UiO : Department of Informatics
University of Oslo

Health Information Systems in Rwanda

Sustainability, Open Source Software, and Android

Simen Skogly Russnes
Master's Thesis Spring 2014

Health Information Systems in Rwanda

Simen Skogly Russnes

20th May 2014

Abstract

A common goal for Health Information Systems in developing countries that are supported by outside actors is eventual self-sustainability. Sustainability is based on the fulfillment of needs in both financial and human capacity which are often lacking in low resource contexts.

In this thesis I show how using Open Source softwares such as the DHIS 2 can improve sustainability as it is both free of charge, and as it encourages local capacity building in administration, maintenance, and use. This has been made possible by working with key actors of the HIS in Rwanda and with other important participants of the global DHIS 2 community. I use the methods of Action Research and Participatory Design, and I take part in projects related to the computer systems of the HIS, one of which being the development of an Android application with the goal of enabling easier data accessibility for those in analytical positions as users of the DHIS 2 software.

Further, as computer system use in HIS is becoming more and more common, DHIS 2 is naturally not the only supporting computer based system being used. A negative aspect of that is that fragmentations of computer systems occur as results of independent programs, which has been the cause of duplications of data and redundant work efforts. Through efforts done at the Ministry of Health in Rwanda I show how merging and reducing systems, and creating automatic interoperability applications for synchronization can reduce such problems and can make the HIS more effective.

Contents

List of Figures

List of Tables

Acknowledgements

I would like to thank both my supervisors for their advice and support, the Ministry of Health in Rwanda for making it possible to participate in their Health Information System, USAID for the funding, and my fellow master's student companion who accompanied me for some of the field work. I also want to thank the DHIS 2 community for their support, and the rest of the HISP team who have given me guidance throughout the thesis period. Finally I want to thank my family for their support in making this thesis possible, and the Norwegian government for providing the opportunity for free education for myself and everyone else in this country.

Chapter 1

Introduction

Health Information Systems in developing countries are gradually becoming more and more computer supported[15]. Sustainability and capacity are key topics for having a successful HIS which can support the country in the long run[23]. However, such HIS are often built up of independent programs by different foreign actors, each with narrow focus areas[10, p. 2]. The result of that can be a HIS with a fragmentation of computer systems that work independently of each other. Such fragmentation of software systems have led to problems like duplicate data and redundant work in the past[31]. In this thesis I wanted to find out how different approaches to computer systems in HIS affect sustainability and capacity building in such contexts. Specifically I focused on the Rwandan HIS, and looked at what particular software system they were using as a case example.

Further I wanted to find out if and how Open Source software can help address these problems and how they can contribute to sustainability. I wanted to see how the Open Source software systems and communities, and proprietary Software as a Service affect HIS differently by conducting research at the Ministry of Health in Rwanda. In addition, I investigated solutions to the challenges in the HIS of Rwanda, and if and how Open Source software and communities are beneficial tools which can help solve them.

Furthermore I developed and tested an Android Application[1] which works as an extension of the DHIS 2 software. The field work case study of Rwandas HIS was an opportunity to experience and work in a context in which to see if and how such an application can be used by different actors by following the Participatory Design principles. Since DHIS 2 and HIS in developing countries usually exists in a low resource context, the target users for an Android application may be limited since the Android devices are relatively costly. The most likely users to have such devices are those in higher up positions, which is why the focus of the application is on analytical functionality. Making it possible to access data quickly and on the go is something that had been proposed as a need in some contexts of DHIS 2 users. Using the methodology of Participatory Design and taking advantage of the situation where I could work closely with target

[1] http://developer.android.com/about

users was a good opportunity for a chance of better meeting the needs of the intended users. The application was released fully Open Source, and while developing it I worked alongside the main developers of DHIS 2. By participating in the DHIS 2 community I got to influence the development of the main system by requesting features that connect the application with the rest of the system. I am also provided with feedback from users of the application around the world through the DHIS 2 mailing lists.

1.1 Motivation

When I graduated with a B. Sc. in Computer Engineering from the Buskerud University College I could have started working as a programmer. I did however find myself too interested with large scale information systems and complex challenges related to them, and would rather work with something in that field than in low scale programming problem solving. After I was accepted into the master's program Informatics: Programming and Networks at the University of Oslo I was able to take courses relating to challenges with Information Infrastructures[2] and distributed systems, and also Open Source software. It was then I got really interested in the topics and found out about the research group Global Infrastructures (GI)[2]. They seemed to be doing interesting work developing computer systems across continents, often in low resource contexts. I got introduced to a new way of thinking, often how to do more with less, where the use of Open Source software was central and frequent.

For project topics I thought it would be interesting to look at how HIS in low resource countries are affecting development locally. Questions like capacity and funding are central, and looking at how different approaches offer solutions to sustainability in such contexts is something I found to be fitting for thesis topics. Whether or not Open Source software is contributing to capacity building is something I wanted to look into.

As many of the projects GI are involved with take place in countries in Africa and Asia it was beneficial, if not necessary to go on a field trip to get some hands on experience with some of the challenges.

For that reason I went to Rwanda for 5 weeks, where the DHIS 2 software was used for implementing the main Health Management Information System. There I worked at the Ministry of Health in Kigali, participating in projects within their HIS, which gave valuable insight into the situation and challenges of their HIS, and provided an opportunity for using the method of Participatory Design during the development of the Android application.

1.1.1 Deciding on Rwanda

Together with my supervisor and co-supervisor I was looking for possible projects, and for a while a project on implementing a regional database for the West African Health Organization seemed like a good candidate. While

[2]http://www.mn.uio.no/ifi/english/research/groups/gi/

looking for projects I participated in some discussions at the university and through e-mail and Skype, and did some brief implementation work. We were looking at possibilites of going to Sierra Leone or the Gambia, but suddenly a project was presented by people at the Ministry of Health (MoH) in Rwanda. At the same time at a training session for DHIS 2 developers at the University of Oslo in October 2013 I had agreed to travel and work together with a master's student from the Norwegian University of Science and Technology who was studying in the same field as me. The university asked the MoH in Rwanda if they would like two master's students to come down and work with them, and together with the company Management Sciences for Health (MSH) who were working closely with the MoH we agreed that we would work with them as interns.

On this trip I would be able to see how the Rwandan Health Information System worked and what computer systems they used. Further I got to work with and experience a context in which DHIS 2 is being used as one of the main parts of a HIS by working alongside some of the main staff in problem solving and project work.

1.1.2 Making the Android Application

Developing apps for Android had been a hobby of mine for the last four years, and I thought it would be beneficial to apply my skills in the field to the DHIS 2 context. The goal of developing the application was to make it easier for DHIS 2 users in analytical positions to access data. To have the data readily available with one click is easier than having to access the data through a web browser, which is the case with DHIS 2 in general. My hope was that it would increase data use as a result of easier access.

The way the app was initially started was during a course in Open Source software development held by the HISP at the University of Oslo. The functionality of the app was based on the requests of one of the main DHIS 2 developers, who thought the application should target users in analytical positions. Only a simple prototype was finished however, but after the course interests from actual DHIS 2 users suggested it would be a good idea to continue on it. In an email from an involved organization (Akros Global Health[3]) it was stated that "We want to give tablets to chiefdoms here in Zambia: the chiefs are not tech savvy, and even something simple like logging in through the browser might prevent system usage." which further went on with their hope that an app would enable easier access to data "so that all they had to do was power up the device and current indicators would be immediately visible, that would be perfect.".

As it was at the same time as I was going to Rwanda where I would get a chance to work closely with other potential users it was a good opportunity to develop a working version which could benefit from the use of the Participatory Design methodology.

[3]http://akros.com/

1.2 Research Objectives

In this thesis I focus on Health Information Systems in developing and low resource countries, and Open Source software, with a specific focus on DHIS 2.

A common concern in developing country projects is how sustainable the projects are and what capacities that are there for the long run. Since DHIS 2 is intended for use in low resource and developing countries, I wanted to look at how the software and Open Source software in general affects development in said context.

As it was previously stated, HIS are often built up of independent vertical programs[10, p. 2]. I wanted to look at some of the most common problems that are faced when HIS consist of many different computer based systems, which is a result of this, and I wanted to investigate how using Open Source software can help solve some of those problems. Moreover I wanted to study if and how the computer based HIS work together, and look at possibilities for improving such interoperability and whether or not it is going to improve the HIS as a whole.

While working in the HIS context in Rwanda I at the same time developed an Android application for analysis purposes in DHIS 2. I wanted to find out if users could benefit from such an application, and how, and wanted to use Participatory Design as a method to achieve better results. By being situated in the heart of a country where the intended users were present I was able to more closely interact with the users during the development. In addition to working with the end users and get feedback from them, I myself first hand witnessed the context in which the app was going to be used, as I figured out how the HIS works, who were part of it, and in what situations it could potentially be used.

To narrow down the field of study I will more specifically focus on the following research topics:

- Find out how the HIS in Rwanda works and how it relates to development challenges such as capacity and sustainability.
- Investigate how Open Source software can be beneficial in the developing country HIS context with regards to lower costs and local capacity building.
- Look at challenges of fragmentation of computer systems and look at possible solutions.
- Find out if and how an Android application for analysis purposes in DHIS 2 is applicable and useful in the low resource country HIS context and see how the Participatory Design methodology can contribute to making the application better.

To make this possible, I made a field trip to Rwanda for 5 weeks in the fall of 2013 where I worked closely with the staff responsible for the Health Management Information Systems (HMIS) at the Ministry of Health in Kigali. There I conducted interviews with staff, went to

the field and observed training of health staff, attended staff meetings, while I identify possible challenges related to their setup. Further I work on implementation and development projects requested by the MoH to improve elements of the HIS, and do rapid prototyping with HIS staff. One of the projects that I worked on was developing an Android application extending DHIS 2, and working in this context allowed me to use the Participatory Design methodology where I could work closely with the intended users during the development stages. Through rapid prototyping and user participation I was able to better meet the requirements of the needs of the users.

1.3 Structure of the Thesis

In the coming chapters I will first describe the theoretical framework within which I will do my work, after which I go into detail on background information on both Rwanda and the HISP and DHIS 2 community. Further I describe the methods I use during the work which includes both Action Research and Participatory Design. Further, in chapter 5 I describe the HIS in Rwanda and some of the current developments that are taking place there, after which I present the process and experience of developing the Android application used for analysis in DHIS 2. Then I discuss my findings with the related research and the presented research topics, before I finally conclude.

Chapter 2

Literature Review

In this chapter I present literature that is relevant to the thesis. I first go into defining Health Information Systems and some of their challenges, before going into the topics of interoperability and Open Source software.

2.1 Defining Health Information Systems

Health Information Systems and their related computer systems are the main focus of this thesis. Here I present a definition of both.

2.1.1 Information System

To define what a Health Information System is, I will first define what an Information System is. [47, p. 26–27] describes it as the part of an institution taking care of the control and maintenance, including the storage of information. They further explain that it is socio-technical, meaning that it comprises of both the technical and human actors are included in an Information System. That means that the staff, as well as the information itself, along with the methods that are used for management. The part of the Information System that runs on a computer can be called a computer based Information System. Such computer based Information Systems will be the most central part of this thesis.

2.1.2 Health Information Systems

From the definition of an Information System, an Health Information System (HIS) can then be described as an IS in the area of health care[47, p. 33]. Typical human actors such as medical doctors and other health workers in addition to their procedures and methods in data collection and handling all go under the term HIS, along with the collected data itself either in paper or electronic form[32]. In [32] they describe the main function of a HIS to be ".. to support decision-making by helping to identify areas for action, setting priorities and evaluating the results of the decision made.". In this thesis when talking about HIS, it includes all aspects of health information within a country, including

both human and non-human actors. However, the computer based Health Information System is the most interesting part of the HIS in this thesis as I focus on challenges related to them. Examples of computer based Health Information Systems are typically computer systems used for the management of health information, such as statistical data tools or electronic patient records.

2.1.3 Health Information Systems in Developing Countries

HIS in Developing Countries follow the same principles as HIS in general, except with challenges such as lacking infrastructures and resources. In the context of this thesis infrastructures such as Internet connections and stable power are most relevant. There is also the question of the local capacity of skilled health workers who are able to maintain and run a HIS. Again, in the context of this thesis health workers with knowledge in IT is essential as most of the focus here is on computer systems in HIS. Local capacity is essential for the sustainability of a HIS[23], especially considering that HIS efforts are often supported by outside actors in the beginning, with a goal of eventually becoming self sustainable. In [21] the authors talk of the necessity of what they call hybrids, who are people who understand both the technical parts of systems, and the more business-oriented aspects. Such hybrids are said to play key roles in implementation phases, and increase success rates.

Building of local capacity is further a way to address the lack of skilled workers. Training of trainers like mentioned in [7] affect sustainability in the way that locals eventually are able to teach themselves, and effectively create local capacity on their own.

2.1.4 Health Information System Challenges

Now that HIS have been defined, I will identify some common challenges related to them.

In [31] they state that current HIS are sometimes seen as obstacles rather than tools, and they summarize the reasons in the following five points:

- Irrelevance of the information gathered
- Poor quality of data
- Duplication and waste among parallel health information systems
- Lack of timely reporting and feedback
- Poor use of information

Although all five are important topics, the most relevant to this study is the third point relating to duplication between HIS. To go into detail on this particular issue, they further explain that it is seen as a common case that

national reporting systems rarely are results of coordinated efforts. They are rather a collection of individual programmes that have developed their own specialized information systems, many of which focus on specific areas such as a specific disease or management service[31]. Such systems usually exist in addition to the general routine health information system, as the data needed in the specialized programmes is more detailed and specific than the routinely collected data. The end result of this is then that the total of the systems become problematic, and that the people responsible for data collection waste time on collecting overlapping information[1].

2.2 Interoperability, Gateways, and Standards

Large institutions and organizations often use a set of Information Systems (IS) for solving different tasks. These systems are often developed individually of each other with no intention of working together[10, p. 2]. This often results in duplicate data entries, and non-optimal work situations. An example of such duplicate data entry is having to update the same information in two seperate systems, for instance a personal profile. Non optimal work situations can mean that people are forced to work using an array of different IS for performing their tasks, which can feel quite tedious and unecessarily complex[34].

To solve problems like these, it is possible to implement gateways acting as connectors between applications[17]. Hanseth defines a gateway and its tasks as follows: "A gateway can be defined in general terms as a link between different elements. Within computer or telecommunication it is used to denote an object linking two different networks or different communication protocols or standards. It is often used to denote a converter or translator between different formats."[17].

Gateways vary greatly in scale, and come in small and large complexities. They could simply come in the form of connecting two Information Systems together by sending data from one end to the other. Though it is a small task, it is important and it could in practice turn many smaller systems into one large system. Using gateways embrace the fact that some IS and their respective computer systems are developed independently of each other, with no original intention of communication with the outside world. Instead of having to develop one large system taking into account all aspects and all problems of a facility, gateways can make use of existing IS with specific functions tailored for their individual use and combine them to become a larger functioning entity consisting of indidividually working systems[2, p. 69–70].

What is meant by interoperability is the ability for systems to interconnect. There are many levels of interoperability, like for example the use of standards that provide a common format for data exchange. [41, p. 18]. Although standards are generally looked at as something formal and massive such as the HTML standard, one could argue that any data format could potentially be a standard. For example, if someone decides to develop an application that has an underlying database, and provides functionality for

that system to export some sort of data report, the data would be formatted using the standard the developers decided to use. If others decided to adopt this way of formatting data, perhaps because they want to use the data reports, it would if enough actors adopted it become a de facto standard[19, p. 411][18, p. 187–188].

As Hanseth states in [17], gateways complement standards. Although it is possible to adopt standards by different parties, it will not always be so, and it will often be a progress over time. Developing a simple gateway can solve the problems of multiple standards without having to do any work on the existing systems, by acting as an intermediary. In many cases this is more practical than having programmers make updates to their software and then update the running software on computers that are in use.

2.3 Free and Open Source software versus Proprietary software

Source code is the human readable format in which a piece of software is initially developed by a single person or a group of people, before it is compiled into a machine-readable format which is run on a computer. Source code can be divided into two forms: Free and Open Source, and Proprietary. Free and Open Source software is software which source code is openly accessibly to anyone. This means that anybody anywhere can download and modify, and use the code however they want, depending on the license of the project. If someone somewhere feels that they can improve the software, they can easily do so, and then upload their changes to the web[36]. This allows developers to make changes they find necessary themselves, without having to rely on the people owning the code.

Open Source projects are as a result often cooperatively developed by people in a global context[46]. While one developer could be located in Norway, another could for example be located in Vietnam without anyone really noticing the location difference. A common case in the Open Source world is that someone develops a piece of software for his own use, and then decides to upload it to the internet for others to also use it. The original developer might even drop out and stop developing the code as it is sufficient for his needs, but other may pick up where he left off, using the code and making necessary additions to it.

Basically there are two ways to upload changes. The first is to upload your changes to the existing project you made your changes to. This requires other developers of the same software to accept your changes, before you can make the addition. The other is called “forking”, which means that you create your own project, creating a fork in the road of the software development process of the given code.

Looking at the redistribution aspects of such software [40] states that “The Open Source movement relies upon licences, copyrigt and patent law to enable the use of source code by others, specifically its modification and redistribution.” They further explain that use can mean different things based on the situation, such as copying the code or application, but

the licences often focus on modification and redistribution. The licences vary where some allow the user to modify and redistribute however they want, where others require the derived code to match the original rights throughout the chain of distribution[40]. The latter is called copy-left, and may for example restrict users to take code that was orginally free, and sell it as closed source. An example of a license is the GNU General Public License (GPL)[1], which is one of the most common Open Source licenses which was originally introduced for "assuring that once software was published under that license, the availability of the source code is guaranteed for all future enhancements"[22]. Proprietary Software is software that is owned, developed and managed by a company or individual. It is usually not free, and the source code is usually not available to anyone except the original developers. This means that the only changes that can be made to the software is to be done by the source code owners[35]. Furthermore, the companies delivering such proprietary solutions often deliver whole solutions called Software as a Service which includes costs of maintenance, support, and servers. The quality of such software is often seen as higher, which is natural as the producers are able to make money off of their product.

[1]https://www.gnu.org/copyleft/gpl.html

Chapter 3

Background

In this chapter I present background information that is relevant to the thesis. I first present the HISP and DHIS 2 projects, and then go on to give a brief introduction to the country of Rwanda.

3.1 HISP and DHIS 2

As a student at the University of Oslo, the Health Information System Programme and the District Health Information Software 2 are interesting topics with relatively easily accessible involved people. The projects are closely linked and were both started by researchers at the university. I have been able to talk with the main developers and implementors on a regular basis which has helped me with valuable insight on different topics concerning my thesis.

3.1.1 HISP

The Health Information System Programme (HISP) is a global network that was established in South Africa in 1994 by researchers at the University of Oslo after the end of apartheid[1]. It is an initiative for creating Health Information Systems mainly in developing countries for collecting aggregate, and other health related data. The HISP have been aiming to help empower disadvantaged communities through the use of ICTs. The programme follow a Participatory Design approach and mainly focus on the development of, and implementation and sustaining of HIS.

The HISP team has created (and is still creating) a software system called the District Health Information Software 2 (DHIS 2) which has the main function of enabling the creation and tailoring of individual countries' needs in aggregate data collection[2]. In short it allows you to create your own HIS without having to program. Although DHIS 2 is created with the intention of making Health Information Systems, it could be used to make any other aggregate data collection application. What's special about DHIS 2 is that it is specifically tailored for working with low resource contexts.

[1] http://www.mn.uio.no/ifi/english/research/networks/hisp
[2] http://www.dhis2.org

It allows for easy collaboration in software development as there are no problems with licensing due to its Free and Open Source software nature. Because Open Source software is usually free of charge, it allows people and organizations with limited resources to take advantage of it.

As many as 30 countries are now using DHIS 2 as their main HIS.

3.1.2 DHIS 2

DHIS 2 is the District Health Information Software 2. It is an Open Source software project that is being developed by people at the University of Oslo, and software development teams in South Africa, India, Vietnam, and various other developers around the world. Since it is Open Source, anyone who wants can download and contribute to the code. DHIS 2 is a generic software tool which enables its users to design and implement a web based health information computer system for aggregate data collection and analysis, along with other functionality.

Since the DHIS 2 development is led by the University of Oslo, and because of my relationship with the University and the research group Global Infrastructures there, a lot of my focus is directed at the use of the DHIS 2 software. Here I go into brief detail of the software, and the history behind it.

DHIS version 1

DHIS 2's predecessor was naturally DHIS, a system developed mainly in Microsoft Office by two researchers from the University of Oslo, starting in 1997[8]. It was developed in post-apartheid South Africa and focused on user participation in its design. The main goal and principle was to empower the local districts in decentralizing and shifting data and information control and flow from a centralized to a more local approach[3].

The principle of DHIS was that it should be as flexible as possible. Since it was with the idea in mind to support a decentralized structure of health services it was essential that the data structure was changeable rather than rigid. The users were to be able to design their own system by designing and changing their own data elements, Health Facilities and organizational units[8]. This way, the users at the local levels were able to control the structure of their own HIS. This again would result in better end results as the varied nature of South Africa in the first place made it difficult to have one standard for the whole country.

The software was developed with the goal of collecting data for local action, and was therefore attempting to participate with the users as much as possible. As the developers were working closely with the end users, it was possible to respond to local requirements quickly. The users were encouraged to report bugs and feedback through informal channels, either personally or through support channels and DHIS implementors, regardless of their hierarchical position. Because they were trying to

[3]http://www.mn.uio.no/ifi/english/research/networks/hisp/

respond to user requests as fast as possible the software was developed using rapid prototyping, sometimes releasing new updates on a weekly basis.

The first DHIS pilot was tried out in 1998, and kept growing until 2001 when it was deployed across all of South Africa. A problem however was that due to the iterative development process that had been the case, DHIS had ended up with a messy and rigid architecture. This was problematic when trying to implement it in new countries, so by 2004 they had developed a new internationalized version which they called DHIS 1.4, which was completely remodelled. The development had still mainly been by a two-man team, which was no longer possible when trying to respond to the local needs of an increasingly growing international context[8].

Enter Version 2

It was then decided that the whole system needed to be redesigned. DHIS had been relying on the Microsoft Office framework, and there were no software collaboration tools being used since there were only two people working closely together on it. The DHIS 2 project was initiated by the University of Oslo, who selected some of the very best and newest Java-based techonologies with which to build the new software. They set up a project in a popular software collaboration platform called Launchpad[4] which made it easy for anyone anywhere with an internet connection to access the code and collaborate in the development processes. Even with the new fresh start of the software the idea of the DHIS 2 was still the same as with DHIS, with trying to empower local users.

There were some hickups and challenges with the new complexity of DHIS 2, but after a few years the first version of DHIS 2 was finished, and was implemented in Kerala, India in 2006[5]. Today it is employed in 30 countries[6].

DHIS 2 is now fully web-based, and runs on a single server for each country. Users access the system through a web browser, where they have personal user credentials to log in. Depending on their user roles and rights, they are able to design their own data reports by creating data elements and indicators used for local data collection. The organizational hierarchies are also still designed by the users themselves, making it easy to add new Health Facilities such as for example a clinic. Since it runs on a single server, everyone enter data into the same database, meaning that the system is always up to date. As soon as someone creates a new data report, or adds a Health Facility, everyone using the same instance of DHIS 2 have access to it[8].

That also means that whenever statistical data reports are entered into the system, it will be instantaneously accessible for others to analyze. Data is reported directly into the database by even the lowest levels of users, and when data is entered, the Health Facility they belong to is reported with

[4] http://www.launchpad.net

[5] http://www.mn.uio.no/ifi/english/research/networks/hisp/

[6] http://www.dhis2.org/

it. This enables very detailed data analysis as high ranking users can see data collected for the whole country through various graphical visualization tools, allowing them to see geographically where the different data comes from. Figure 3.1 shows a graphical map plot of ANC registration coverage rate by district in Rwanda with data from their live database. The ANC registration coverage shows the rate of which women register for their first prenatal care visit, as the start of a standard program consisting of 4 visits to their local clinic for checkups related to the pregnancy. The data that is displayed has been collected monthly at the local Health Facilities throughout the country, where the number of registered first visits is compared with the expected number of pregnancies to estimate coverage. This data is then stored as an indicator in the DHIS 2 database, which can later be used to make various graphs, and maps like the one shown here. Since the data is collected and entered at the local clinics, the data has details on where it was collected, making it possible to easily drill down into the map of the country, to get a closer look at where the data came from. Having detailed location based data like this makes it possible to identify problems with areas that are abnormal, with an accuracy down to the Health Facility level.

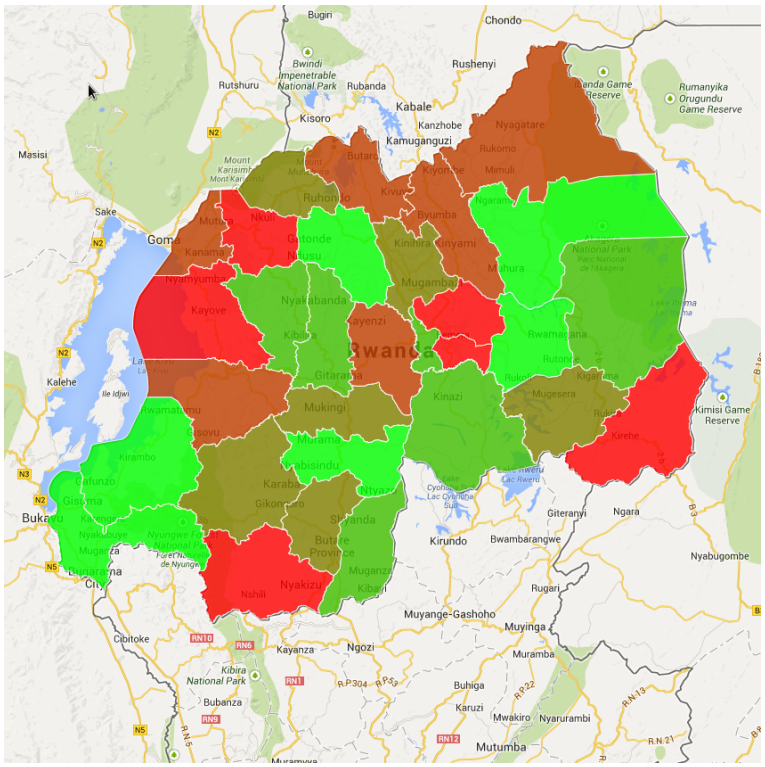


Figure 3.1: Graphical map plot of ANC registration coverage rate by district in Rwanda. Red color indicates low coverage, and green high.

Since many of the users are in low resource contexts, maintaining a stable connection to the internet is sometimes difficult. To handle this

offline support is also included. Although the software is web-based, users can cache DHIS 2 so that it can be used without an internet connection. They can then enter data locally at all times, and connect to the internet when they want to send the reports to the server.

Although the DHIS 2 project was initiated by the University of Oslo, it is now being developed by teams around the world in South Africa, Vietnam, and India as well as by developers in Oslo. By using the Launchpad platform it is easy for users to report bugs, make requests, and ask for support through mailing lists. The worldwide DHIS 2 community has now grown to the extent that users can help each other with support questions.

3.2 Brief history of Rwanda

Rwanda is a small country in the east central part of Africa. It is perhaps most famous for the genocide that occured in 1994 where the Tutsi ethnic minority was slaughtered by the Hutu majority. Around 11% of the population was killed over the course of about 100 days[13]. The country has in later years developed with incredible growth in both political and socio-economic progress, with president Paul Kagame being elected in 2003, and re-elected in 2010[39].

3.2.1 IT in Rwanda

Since the country is quite small, it has been relatively easy to provide internet access throughout the country. The use of USB modems is the most common case, which gives people internet access even in villages in the countryside. The reason for the popularity of the USB modems is related to the lack of landlines, and the fact that there are frequent power cuts which makes it hard to maintain a stable internet connection using a router dependent on electricity. Because of the size of the country, it has also been possible to deploy DHIS 2 countrywide in less than two years without too many complications.

Figure 3.2: Map of Africa with coloring of Rwanda to show the size of the country relative to neighbouring ones.

Chapter 4

Methodology

In this chapter I explain the different approaches and methods I have made use of during the period of this thesis. Much of the thesis is focused around field work, and I use the method of Action Research presented in the Canonical Action Research theory and I here describe the theory behind it. Participatory Design has also been used for collaborating with users during project work in the HIS and when developing the Android application. While conducting the research I do interviews with key staff, and participate in e-mail discussions and other electronic communication technologies like Skype, Google hangouts, and Facebook. I further go on to explain the projects I participated in with the MoH Rwanda, and explain the basics of the process of developing the Android app for analysis in DHIS 2.

4.1 Field work

As it was previously stated I went on a field trip to Rwanda for 5 weeks for collecting data. This was the largest portion of field work related to my thesis, and while there I did a case study of the HIS in Rwanda to understand challenges relating to their computer based systems with regards to sustainability, while at the same time working on specific development projects that were presented by the MoH.

Landing on the projects in Rwanda was not that simple however. I knew, and had agreed with my supervisor that it would be a good idea to get some hands on experience by doing a field trip case study of DHIS 2 being used in a country, preferably somewhere in Africa. Before deciding on going to Rwanda there was a lot of back and forth about different projects that I could possibly participate in for my thesis, but as an interesting project was presented by the MoH in Rwanda it was decided to be a good opportunity.

While in Rwanda I was working based at the offices of the Ministry of Health. I had close contact with the HMIS team, and participated in meetings, discussions, and training sessions with them in addition to helping with HMIS management and improvements. We also went to visit other offices and branches in the government and in the health sector for discussions related to projects in the HIS. For the first four weeks I worked

together with another master's student from Norway, until he left and I proceeded to work alone together with the HMIS team. Working closely with the MoH staff allowed me to get a good overview of the systems that were being used, and I could take part in discussions on how to make improvements. The same master's student returned to Rwanda a few months later, and we were able to communicate and reflect on the progress that had been made using Skype and Facebook.

To explain the work I was doing it can be divided into three parts: 1) Getting an overview of the HIS by conducting interviews, taking part in discussions and e-mail discussions, reading reports, and presentations. 2) Working on development projects within the HIS. 3) Developing and testing an Android application which allows for easier analysis of DHIS 2 data in a smart phone and tablet context.

4.1.1 Getting an overview of the HIS

It was important to get to know the structure of the HIS in Rwanda, for a better comprehension of the context of the conducted work.

Getting to understand the Rwanda HIS was not a straight forward process. Since nobody knew everything, the information had to come from different sources and in different ways. Much of the information I got was from the HMIS team at the MoH, whom I worked closely with for the whole of my stay. I was given a few presentations about the systems to give me a brief idea, but much of the information was gathered in different unstructured discussions, meetings, and e-mails, different interviews with MoH staff, and general asking around. It would often be troublesome to find the right person to ask, and ask them the right questions, but it was not impossible.

As there was no single person responsible for all systems I had to conduct interviews with people working with the different systems to get the information I needed.

4.1.2 Development and implementation projects

While gathering data to understand the HIS I was at the same time working on three development projects the MoH had presented. These projects helped me understand some of the aspects of the HIS through practice, and some of the problems they were facing. The projects are described as follows:

Malaria Surveilance implementation in DHIS 2 This project was about implementing an SMS based solution of reporting data to an existing DHIS 2 instance. The project was divided into two parts, one in which the goal was to gather information surrounding new cases of malaria for identifying the source, the other about reporting statistical weather data three times a day at selected Health Facilities.

Interoperability of computer based HIS in Rwanda The MoH were using an instance of DHIS 2 as a central Data warehouse where they

would gather data from all the other various systems. They would push data into this database once a month, but the problem was that they had to manually run a set of self made SQL queries to pull the data from one end, and pushing it into the Data warehouse. This was a tedious process, and they wanted it to somehow be automated, or at least semi-automated. Furtermore, they had to manually synchronize organization units and other elements whenever a new one was entered into one of the databases. It would also be beneficial if that process was automated.

Migration of disease system TracNet to DHIS 2 A system based on Interactive Voice Response (IVR) named TracNet was being used at the time. This allowed health workers to call in and report via their phone and by wording out simple commands to an automated server. This system was however quite expensive, both because they had to pay for the minutes each health worker spent on the phone, but also because the software was proprietary and any maintenance or upgrades would have to performed by the company Voxiva[1] who initially made the system. For those reasons they wanted to implement the functionality in DHIS 2 instead. The proposed solution was to use SMS reporting to solve this problem as well.

For the interoperability project I looked at ways to implement an automatic integration engine for synchronization between the systems, while at the same time identifying in detail the system roles, and also the possibilities they offered in ways of exporting and importing data. I learned that a Ph. D. student who was highly involved in the development of DHIS 2 and the implementation in Rwanda had been working on something like what we were trying to accomplish, so we joined forces. As he was situated in Dublin, we communicated over Skype and were able to collaborate using development platforms common in the Open Source world. The development was led by the Ph. D. student, while I assisted. Up to the end of my stay something usable was finished, but it was decided that the framework being used was not optimal and a transition was made. This was not very complicated, and the collaboration process continued as I returned.

For the other two projects I participated in meetings and investigated ways to implement SMS functionality regarding the Malaria Surveilance problem, and the migration of TracNet into DHIS 2. We were looking at ways to implement solutions using SMS, and it was decided that the best way to do something like that was to use the SMPP protocol. A master's student from the University of Oslo had been working on a mobile project in DHIS 2 in Rwanda the year before so we picked up some of his work[27]. Some of the challenging parts of the process on the technical side was to identify how to implement the solution, and to set up the appropriate subscriptions and agreements with telecom operators.

[1]https://www.voxiva.com/

Further I participated in e-mail discussions and meetings with the people responsible for the malaria program at the National Institute of Statistics Rwanda. The malaria surveilance project was quite large and was divided into two parts. One reporting weather data three times a day at selected sites in the country with installed monitoring equipment, and one investigating the surroundings of malaria cases. The first case was decided to be solved using SMS since the amount of data to be reported was relatively small, and the other case was to be implemented using web forms as the data required would be too substantial for a single SMS. Since the HMIS team at the MoH had years of experience implementing reporting forms in DHIS 2, they were assigned the task, but I did try to make some suggestions based on the data that needed to be reported.

4.1.3 Field notes and Observation

During meetings and field trips and other relevant situations to the study I kept a notebook to keep track of everything that happened. Since the days get busy and many things happen it is naturally important to write down what happens to not forget important notes. The same thing is true for when observations are made, which can also be relevant as a part of a field study. I kept a journal of the work that was being done and transcribed the notes every night at the end of the day as a reference to look back on to write this thesis. Keeping a journal is something that was suggested to me by one of my supervisors, who had just finished his own Master's thesis when I started, and had continued with a Ph. D.

4.1.4 Interviews

During my stay in Rwanda I conducted interviews with staff related to the Health Information System. Some of the interviews were structured in the way that I prepared a set of questions related to HIS, and took notes and recorded using my smart phone during the interviews, which I later transcribed. These interviews were with people working at the MoH in Kigali who had connections with the HIS work in the country. Some other interviews were more open, where I interviewed data managers during the breaks at a training workshop in Kayonza.

I also conducted interviews with implementors and developers from the OpenMRS team to find out what their system was doing and how much of the country was using it, and how it could be possible to integrate it into the interoperability engine.

4.1.5 Developing and testing a DHIS 2 Analysis Android application

A big part of my contribution was that I developed an Android application to be used for DHIS 2 Analysis purposes. The app was inspired by the existing functionality of DHIS 2, and some modules were implemented to be run natively in a mobile Android context. The Application was developed

fully Open Source and it was encouraged that other actors pick up the work and support the application development in the future. While developing the application I was able to participate closely in the DHIS 2 Open Source community, and experience first hand how it is possible to influence the evolution of an Open Source software.

The most interesting part about the Android application is whether or not it would be useful for the users, and who could actually use it. Due to the nature of DHIS 2 typically being present in contexts of lower resources, the higher up actors were the target users, which is why it has an emphasis on analytical functionality. The goal of the application was to make it easier to access data and be able to analyze without having to be on a computer, but rather be able to open an application on the phone with a single tap on a screen.

Being able to participate in a country context where DHIS 2 is being used, and to get to personally know the target users was valuable and beneficial for understanding their needs. This helped improve the application as a whole as it was possible to respond to user feedback directly.

The application was developed both before, during, and after the trip to Rwanda. Before going I was able to discuss with the main DHIS 2 developers on what features to implement, while at the same time being able to suggest additions to the main DHIS 2 software which supported the application. Since the application connects to a given running DHIS 2 instance, it is important that the app can connect and retrieve the needed data. In the beginning some of the required functionality did not exist, but by talking to the other developers it was implemented quickly, and is now a part of the software which is available to all the users.

While in the country I was able to get feedback from some of the intended users so that it was possible to respond to suggestions immediately. Typically I would show the application to the HMIS staff at the MoH, who gave me their views, after which I could implement changes promptly, and show it again the next day.

After the stay I published the application to the world using the Google Play portal, making it possible to download the application for anyone who has an Android device. I could then see statistics on where the app was downloaded, get further feedback from the users, and look at more improvements that could be made. I kept contact with the key staff at the Ministry of Health in Rwanda, who could directly tell me their experiences with the application.

4.2 Action Research

For studying a case of a HIS in a real context, I use the research method Action Research. Action Research is however somewhat hard to define as it is viewed differently by different people. Jean McNiff states that it is a “particular way of looking at your practice”. He further states that “research is always about creating new knowledge”, and that in Action Research the

knowledge is created by, and about the practice[33].

Checkland and Holwell explain it as a close collaboration process between researchers and people in the studied situation allowing hands on experience with the topic in question[12].

In this particular case however, I will base my process of Action Research on the theory of Canonical Action Research (CAR)[14]. The Canonical Action Research proposes five principles to follow for the best results in addressing organizational problems while at the same time conducting research which can benefit academic communities:

- The Principle of the Researcher-Client Agreement (RCA)
- The Principle of the Cyclical Process Model (CPM)
- The Principle of Theory
- The Principle of Changing through Action
- The Principle of Learning through Reflection

They elaborate on the five principles as follows: The first principle, the principle of RCA suggests that before doing anything else, both the researcher and the client must agree on the use of CAR. Further, the client and the researcher should come to a mutual agreement of what is to be done, and collaborate on defining plans and goals. This is to strengthen trust and validity of the research, and to ".. promote a spirit of shared inquiry"[14].

The second principle, CPM, proposes five stages of a cycle that a project should go through. These are diagnosis, action planning, intervention, evaluation, and reflection (Figure 4.1). If necessary, additional iterations through the cycle should be made, generally following the stages in the given order, although allowing for jumps between unlinked stages if necessary.

The five stages are meant to help strengthen and make the research be conducted in a systematic matter.

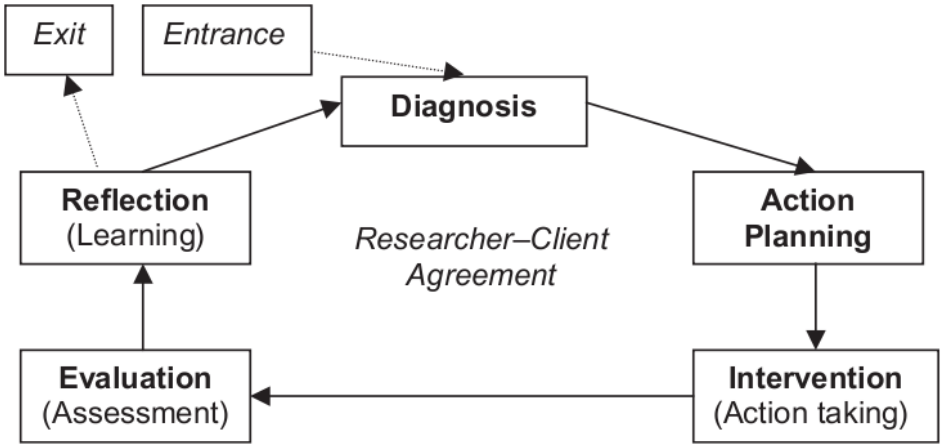


Figure 4.1: Five stages of RCA in CAR theory[14, p. 72]

The Principle of Theory suggests that researchers conducting CAR should do so within a theoretical framework. That is, ".. rely on one or more theories to guide and focus their activities"[14]. To do this, researchers should select certain topics, and review literature that is related to them. This is meant to facilitate and narrow down the field of research, as it is easy to get lost in "a sea of data"[14, p. 74].

The Principle of Changing through Action is the principle that for the CAR to be successful, changes should be made. The whole point of CAR is to take action and try to better the situation which is being assessed. The changes to be made should be guided through the previous stages, and should be well documented and analysed when finished to enable learning and further action.

The final principle, the one of Learning through Reflection is there to help govern the knowledge outcomes of the project. By reflecting on the performed processes the researcher can contribute to the academic community for further work on the same topic, or related work making use of the findings. It is also important in regards to the client whom is usually interested in the practical outcomes of the changes. It is then important to be able to document and argue for what has been done, and how the CAR process has changed the basis on which it began.

In this thesis I base my idea of AR on the five principles presented in the CAR theory, but follow a less strict model to account for uncertainties and changes in situations. More specifically to show how my approach was adapted, it is best to go through how the five presented stages were addressed in practice in comparison to the theory.

The first principle states that both the researcher and the client must agree on the use of CAR. It further states that the client and reseacher must come to an agreement of a definition of plans and goals. In this particular case the client can be seen as the MoH in Rwanda and the researcher as the author. We did not agree on using CAR, and in fact we didn't even discuss CAR at all. We did however define plans and goals together. The plans were the projects that were followed throughout the engagement period both during and after staying in the country.

The second principle of the cyclical process model presents the five stages that should be followed during the projects. The diagnosis were perhaps mostly made by the MoH as they had already identified the problems, while we together did the action planning stage as we investigated possible solutions. This is true for both the interoperability project and for the SMS messaging projects. The interventaion stage (the one of taking action) can be seen as the part of the project where implementations were done, meaning developing the interoperability engine, and working towards a working SMS solution, and the Android application. When it comes to the evaluation stage it could fit into the discussion part of this thesis, as I describe the end result, how it was done, and what were the positive and negative parts of it. The reflection stage is in this case hard to distinguish from the evaluation stage as it also fits best into the discussion part of this thesis, but is more related to the conclusion and research questions, than the actual discussion and description of the performed processes. The principle sug-

gests multiple cycles, and even jumps between the stages, but in this case a single cycle has been performed. It is because of the length of the project being relatively short in my case. The projects themselves are still being carried out by different actors but within the frames of this AR effort it is done.

The way I followed the principle of theory was by using the topics and theory that was presented in the literature review in this thesis. By limiting the theoretical framework to those topics, it has been easier to focus on more specific problems, and not getting overwhelmed with a focus area too large.

Changing through action is in this case very much linked to the intervention stage of the second principle. That means the implementation work that was done during the project, which is described both in the previous chapters and here.

This thesis as a whole, especially considering the discussion and conclusion parts is the way I follow the principle of learning through reflection. Since the thesis documents the processes before, during, and after, including results it is the way the research is finally reflected on and finished.

Although I have used a less strict Action Research model than the CAR theory, it has been a benificial tool which has helped facilitate the research through following the offered guidelines.

4.3 Participatory Design

In the Routledge International Handbook of Participatory Design[38] the authors define Participatory Design as follows:

"a process of investigating, understanding, reflecting upon, establishing, developing, and supporting mutual learning between multiple participants in collective 'reflection-in-action'. The participants typically undertake the two principal roles of users and designers where the designers strive to learn the realities of the users' situation while the users strive to articulate their desired aims and learn appropriate technological means to obtain them. "

This notion of Participatory Design comes from the Scandinavian research projects[16] where it is central that the users of intended technologies are part of the design processes. Users in this case means those who will eventually interact with the system that is to be designed once it is finished. Since it can be challenging for a designer to envision how the system will be used it is beneficial to collaborate with the users during the process. On the other hand, the users may not be able to express their desires as they don't know exactly what they want, and because they don't know what is possible. It is then beneficial for both parties to work together and merge their knowledge of both the technological aspects, and the context in which the system will be used for the best possible outcome. Through prototypes

and various representations of the system, the users can give their impressions and feedback relatively easily without needing knowledge in formal design processes[38, Introduction].

The use of the PD theory was mostly in the development of the Android application, but was a part of the other projects as well as it is some times hard to distinguish between when work was being done on the different projects. This is due to the fact that the HIS project work was a way of understanding the usage context. Looking more closely at the way the use of PD in this thesis was carried out, the following interpretation has been followed. In [43] the authors state that "A key PD principle is to bridge and blur the user-designer distinction from both directions, through mutual learning processes. By this approach, users must be able to engage with the artefact under design, and designers should build a thorough understanding of the life-world(s) of the users, to have a more complete perspective of the system in use.". Building on this notion of PD, the situation of working with the staff at the MoH both in testing and feedback sessions of the application itself, and in situations related to other project work in the HIS has been beneficial. That the developer was able to get to know and understand the context in which the product would be used, including both understanding the HIS in general as a context and also the day to day working lives of the users has resulted in valuable insight. Furthermore for the intended users to be able to directly give feedback on the artefact during development has contributed to positive changes.

The Participatory Design followed a cyclical development process as described in [9] where the authors describe how they in HISP used Participatory Design in what they call “four different but interconnected cyclic development processes.” These processes ranged from the low level aspects, where the lowest level is the Development of software, and the highest level is Action Research cycle. If the project carried out here would be placed within this ranking, it would have to be even more low level and specific than their presented software stage as it is a very specific application area, within the software they were actually developing. Such a specific and contained application area of making an Android application for the use of one particular module within a larger software system has been a good project in which to practice Participatory Design. Due to the small nature of the project, it has been easy to develop rapid prototypes as described in [9] and for end users to influence through testing and feedback the artefact under development.

An important aspect of the development project to add is furthermore how the developer was communicating with two particular groups of actors, and working as an intermediary for the translation of needs between them. These two groups of actors were the intended end users in Rwanda’s HIS, and the other the core back end developers of the main DHIS 2 system. Since the application was developed based on functionality of the existing DHIS 2 system, there were some times additions that needed to be made. These needs were usually identified through working with the end users in Rwanda, after which functionality additions were requested to the core developers, who added the functionality, which then enabled to implement

the originally identified need.

4.4 E-mail, Skype and Google hangouts calls, and Facebook

I have been participating in e-mail discussions regarding topics related to my thesis study. These are sometimes large discussions of different actors in formal settings, or smaller groups of less formal contexts. Most commonly such discussions are related to development of the Open Source software that is being used in Rwanda. The topics are usually regarding finding ways to solve problems and implementing new functionality.

As a tool, Skype and Google Hangouts has also been used for collaborating across continents. Especially when working on software development tasks this has proved useful as screen sharing allows for easy explanation.

Communication using Facebook chat functionality was also beneficial for communicating with participants on an informal basis. The reason why Facebook was used is because it is readily available at any device with a web browser, providing an effective chat client without the need for any installed application.

Chapter 5

The Rwanda HIS

In this section I present the most important parts of the HIS in Rwanda with an emphasis on the DHIS 2 software. This includes the different computer based systems, what their different purposes were, how they were being used, who were using them, and challenges related to them. It was natural to focus on DHIS 2 since it was the most widely deployed software system in the country, and because of my connection with the University of Oslo, which administers the DHIS 2 software through the HISP network. First I describe the structure of the country, the different actors who participate in data collection and analysis, before going into more detail on the actual computer based systems. The main focus is on the use and circumstances surrounding DHIS 2 based systems, with additions of other relevant computer systems.

5.1 Country Structure

To understand the way computer based HIS are being used it is important to understand the underlying structure of the country and how it is divided into levels. The country is divided as follows:

Central level The whole of Rwanda.

Province Rwanda is divided into the provinces East, North, South, West, and Kigali City.

District Each province is divided into districts, where each district has one or more District Administrative Hospital.

Sub district Some districts are divided into sub-districts. Usually 2. These sub districts usually have a District Hospital.

Administrative Sector The districts or sub districts are divided into administrative sectors. Here, there will usually be a Health Center and a mutuelle section, which have authority over their underlying entities.

Cell The cell level is what the administrative sectors are divided into.

Village Lowest level of country division.

5.2 Actors and Users of DHIS 2 in Rwanda

There were different user roles related to DHIS 2 in Rwanda. They range from the very top level administrators responsible for everything from basic training to server setup, down to the lowest level of community health workers.

HMIS Personnel and DHIS 2 Administrators

The people administrating and implementing the use of DHIS 2 worked in a team called Health Management Information System (HMIS) in the MoH. The team consisted of about 8 people, who were responsible for server maintenance, implementations, database administration, user support, and user training for the whole country. One specific task was to design the reporting forms used by the data managers.

These people were experts in DHIS 2 and had been trained in several workshops run by HISP to be the main DHIS 2 staff in Rwanda.

Data Managers

At each Health Center and District Hospital there was at least one person with the title Data Manager. This person was an expert in DHIS 2 and was responsible for everything that had to do with DHIS 2 for that particular facility. The person performed data entry based on paper forms collected and filled out by other health staff. The data manager also typically analyzed data and generated reports through DHIS 2 when other health personnel requested it.

Monitoring and Evaluation Officers

Each Health Facility typically had a monitoring and evaluation officer who analyzes the data in their area. They typically looked for outbreaks, irregularities, or other cases related to the data in the DHIS 2 database.

Community Health Workers

In the lower levels of the health information hierarchy, people from the community usually performed simple procedures and collected data. These people are called Community Health Workers (CHW). There were about 45000 CHW present in Rwanda. The community health workers did not necessarily get paid much, but followed a scheme called Performance Based Financing. The CHW were usually administered by a Head of CHW. The Community Health Workers reported to a data manager at their local Health Center, who entered data into DHIS 2. They also reported using mobile phones into other systems using RapidSMS and TracNet in some

cases, and the plan was that they would enter data directly into DHIS 2 themselves when SMS support had been implemented.

Government Officials

Some people in higher up government positions would typically have access to DHIS 2, where they were able to look at data relevant to their situations. It would be beneficial for them when going abroad to be able to log in themselves and access data directly when they are presenting numbers for example at conferences. Before DHIS 2 they would have to call the HMIS staff and have them look the numbers up for them, but were now able to directly get the information themselves. The Minister of Health had access to DHIS 2 and had a personal dashboard and was able to analyze data from the whole of Rwanda.

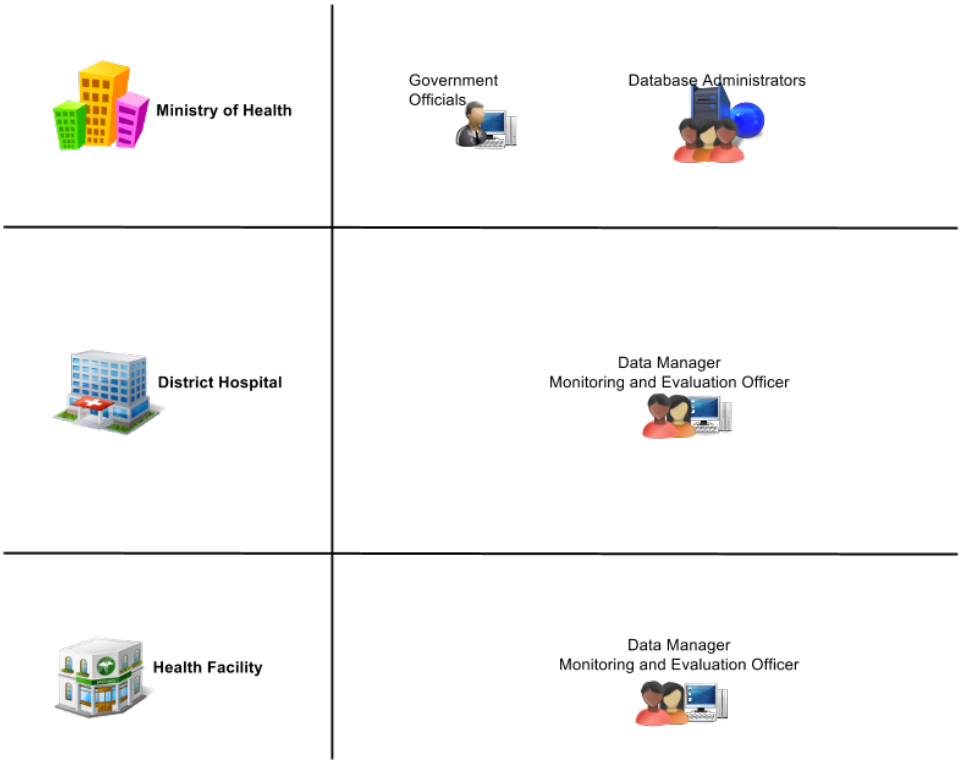


Figure 5.1: Overview of the user roles of DHIS 2 in Rwanda

5.3 How the data is collected

The way data was being collected throughout the country varied from paper based forms, to online data entry, to SMS reporting.

When it comes to data collection into DHIS 2 there were basically two contexts data managers entered data. These were in lower level Health Centers, and in District Hospitals. Health Centers were usually a local clinic, a vaccination station, a pharmacy, or some other facility related to

health work. A District Hospital was one of the top level facilities, and there were usually only one or two per district.

Typically health workers would register data onto paper forms in the field and from visits. These forms had been designed with specific data to be entered into DHIS 2. The data manager responsible for a given facility would routinely collect paper reports from health workers of various disciplines, and enter the data directly into the online DHIS 2 system. At the District Hospitals there was a big monthly form that was divided and passed out to all the different branches, who filled out each their section, after which the data manager entered it into DHIS 2. When the data managers entered data, they entered data directly into the DHIS 2 database, making it immediately available for others to analyze. A big part of the data was routinely meaning that the same data elements were entered on a weekly, monthly, and so on ranging up to five year periods.

The way they entered data in either context was similar, as the data managers entered data collected by other personnel working at the same location. Although the District Hospitals had access to read the information of all underlying HC, the District Hospitals' data managers could only view data in the lower levels, and not enter data other than in the DH.

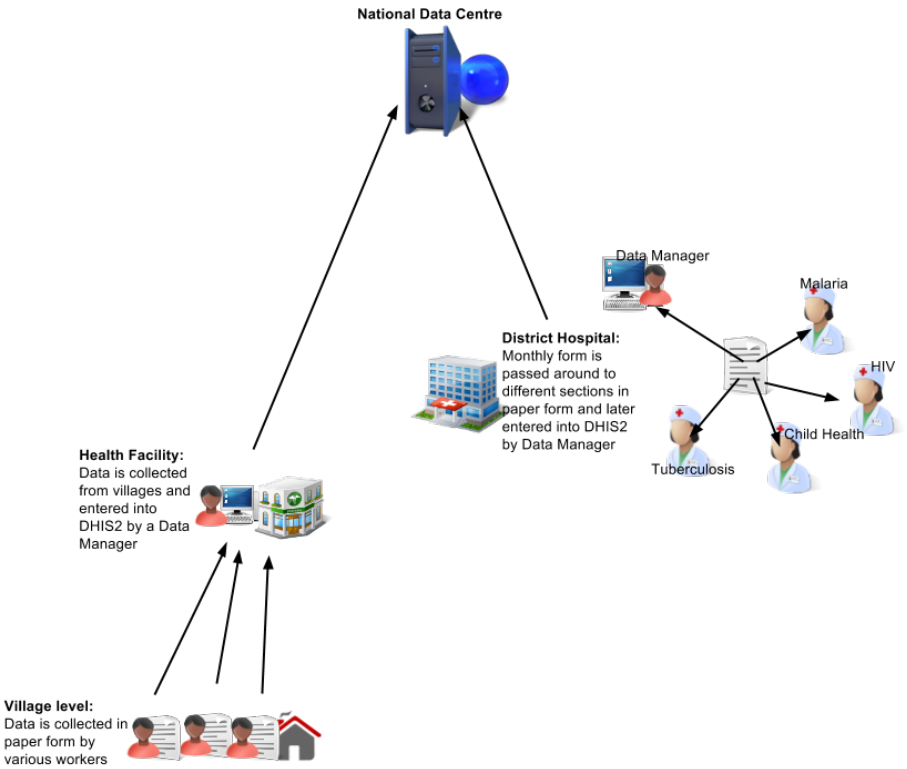


Figure 5.2: Overview of the data entry flow into DHIS 2 in Rwanda information flow

Performance Based Financing (PBF)

In the communities, much of the financing was based on which procedures were performed. CHW from Health Centers would typically report on numbers of performed procedures and get paid based on the amount. The HC they belonged to then got allocated funds, so that the money went to the community rather than to specific actors. This allowed the communities to invest in projects like for instance small dairy farms or motor taxis that could govern them as a whole, rather than giving the funds directly to one person.

Mobile Reporting

Although SMS support was not yet implemented in DHIS 2 in Rwanda, the system RapidSMS was being used. Another system that was being used until recently was the Voxiva TracNet Interactive Voice Response based system which was now being replaced by DHIS 2.

It enabled health workers in remote regions to be able to report data only using a mobile phone directly to a central database.

At the moment there were significant work efforts in progress to implement SMS reporting in DHIS 2 in Rwanda. There were both technical and other challenges in setting up and registering with a short code subscription with the country's main telecom operator (MTN).

5.4 Data analysis

The data entered in DHIS 2 was being analyzed by various actors for decision making and monitoring for outbreaks. Typically a data manager was responsible for data entry as explained in previous sections, and also for generating reports for other staff at their local center or District Hospital. There was a hierarchy of data access, allowing the higher levels to analyze data in the levels under them, but not the other way around (figure 5.3). At the Ministry of Health, users could access data for the whole country. At District Hospitals the users could access their own data, and the data for all the Health Centers in their district. The lower level Health Centers could only access and view their own data.

In addition, each Health Facility had one Monitoring and Evaluation Officer, who checked the data regularly and looked for irregularities and outbreaks.

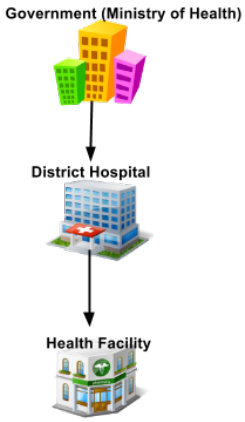


Figure 5.3: Overview of the data access hierarchy in Rwanda information flow

5.4.1 Generating reports

DHIS 2 can get quite complicated and technical, especially for those unfamiliar with the system, so the MoH had decided that each Health Center should have a data manager who was trained to specialize in DHIS 2. Whenever a person working at a Health Facility or District Hospital wanted a report from DHIS 2, he or she requested the given report from the data manager who then proceeded to generate a report. This report could then be printed to paper or be viewed on the computer as is in a PDF or some other format.

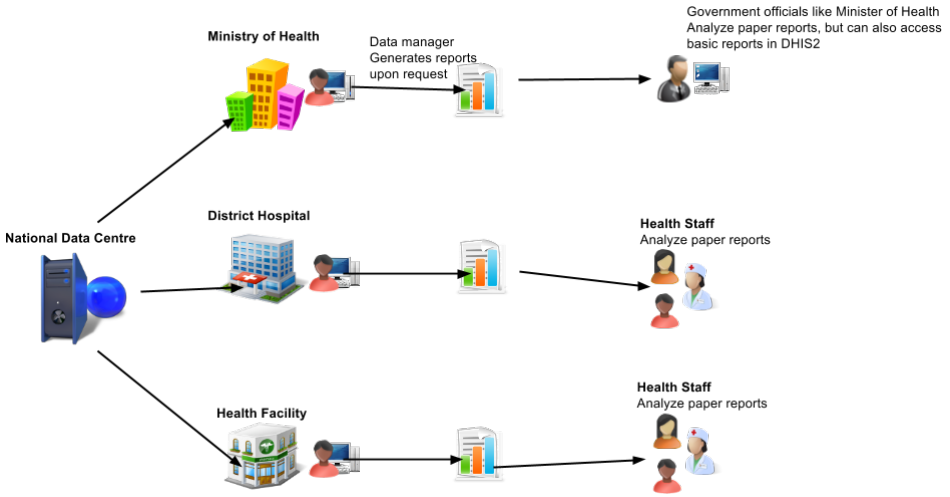


Figure 5.4: Overview of the data analysis flow for report generation information flow

5.5 Computer Based Systems

Now that it has been established how the HIS works on a higher level, I will describe the most central computer based HIS that were being used in the country. The systems were for aggregate data collection and analysis, personal tracking and patient records, registers, human resources, and other programs. I have divided the systems into systems based on the DHIS 2 software on the one hand, and other software systems on the other.

5.5.1 DHIS 2 in Rwanda

In Rwanda they were using DHIS 2 as a big part of their HIS. It was used to implement their main Health Management Information System (HMIS), in addition to other more specific tasks. They used the software mainly with the purpose of aggregate data collection and analysis, but also with tracking individual patients, and also with storing data related to performance based financing. The system was distributed country wide. While visiting Rwanda, DHIS 2 had been used for about 2 years in the country, and had up until recently only been used for data collection and reporting. Since they had been collecting data for two years they were now able to start analyzing and use the data in various charts, pivot tables and reports to help with decision making. Everyone, including the lowest level data managers reported data directly into the DHIS 2 database, making it readily available for analysis in the higher levels immediately. They had divided different tasks that they used it with, and had created 4 different systems using DHIS 2 (figure 5.5). These were the four systems' different tasks:

HMIS

The Health Management Information System was the biggest in terms of database size. It was being used for statistical aggregate data reports that were reported ranging from weekly to five yearly periods. Examples of such data includes Antenatal Care, number of cases of different kinds of malaria, and infant deaths to name a few. The data collection was performed by trained health staff who collected data at the lower levels and in District Hospitals.

Health Finance

They were using a financial scheme called Performance Based Financing (PBF). Here community health workers and Health Centers reported on what kind of procedures they had completed and got compensation based on this. The Health Centres then got money that they could distribute in the community. In one instance they had been able to purchase a set of cows and start a dairy farm.

Individual Records

The Individual Records system was being used for tracking individual patients. As they had recently adopted the individual tracking functionality of DHIS 2 it was only being used for tracking the treatment of tuberculosis patients at the moment. Here they stored information about people's treatments and appointments to be able to follow up individual cases.

Data warehouse

The Data warehouse acted as a central repository of the most important data for analysis in the country. Many of the different computer based HIS, including those that are not DHIS 2, push data to the Data warehouse. The Data warehouse was then used to create charts, tables and reports so that different actors could analyze data and make decisions.

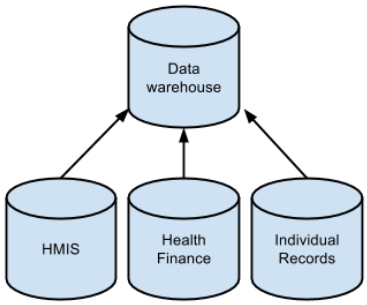


Figure 5.5: DHIS 2 systems currently active in Rwanda, indicating data flow

5.5.2 Non-DHIS 2 Computer Systems

There were various computer based HIS being used in Rwanda's Health Information System in addition to the ones implemented using DHIS 2. Here I describe what I found to be the most central and important ones.

OpenMRS

OpenMRS (Open Medical Records System) is an Open Source software system that enable the customization and tailoring of a medical records software system [1]. They had been piloting it since 2010 in Rwanda and mainly focused on HIV cases, where they tracked patients diagnosed with HIV. At the moment at least 300 Health Centers were using OpenMRS, and at least one District Hospital in each province. At the Ministry of Health they had hired six developers to only work on improving OpenMRS. The system was not web based, and each facility ran their own instance with only their own database as their scope.

[1] http://www.openmrs.org/

iHRIS

iHRIS is a human resource management tool designed to easily allow users to make a health workforce information system. This allows for the mapping of what kind of personnel with various skills are available in the country. It is open source, and was at the time being used by 15 countries, mainly in Africa. The system was being used across the whole of Rwanda, and all Health Facilities used it. It was introduced in Rwanda in 2009.

TracNet

This was a proprietary software system developed by a company called Voxiva. The system was mainly used for tracking data on HIV patients and was deployed across the whole of the country. The Ministry of Health had however decided to phase this system out and re-implement the functionality in DHIS 2, and the plan was to be finished in January 2014 but in May 2014 it was still not done.

RapidSMS

This is an Open Source software framework that allows for the creation of mobile SMS solutions for reporting data. It was deployed nation wide in Rwanda. Currently there were 45000 community health workers each equipped with a mobile handset capable of sending SMS messages. They performed monthly reports, and also tracked all children under the age of five and pregnant women, and sent SMS alerts if an urgent event occurred. The SMS was then automatically also sent to an ambulance, the nearest Health Facility, and to the Ministry of Health for followup purposes.

Health Facility Register

The Health Facility Register was a web based application developed using a platform called InSTEDD. The system acted as a registry of all of the country's Health Facilities. This system had a well equipped WebAPI which allowed for easy access to the data from outside the system.

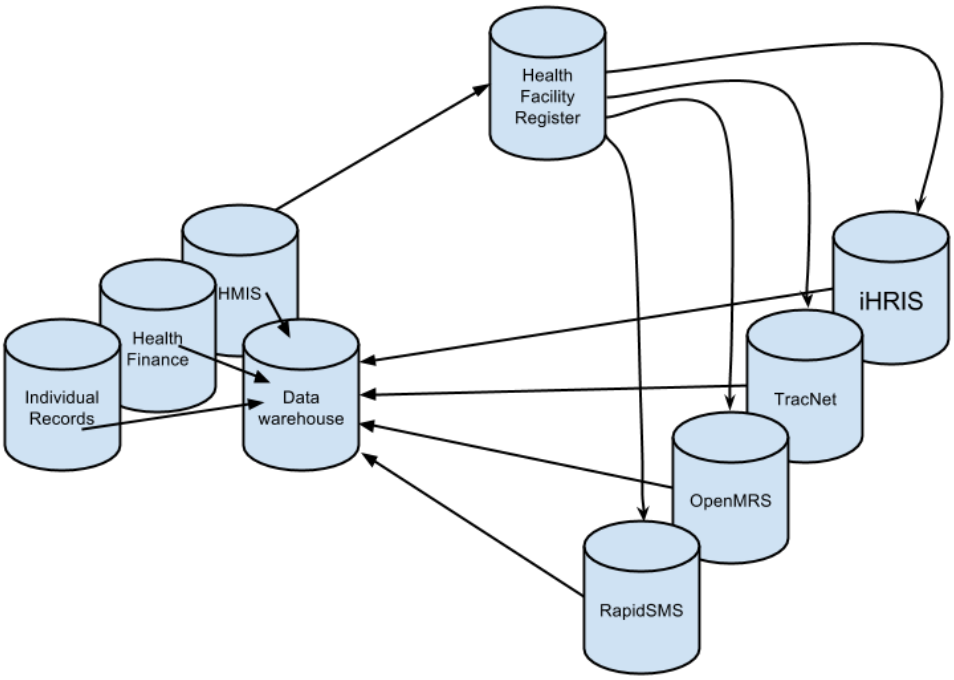


Figure 5.6: Overview of the main HIS present in Rwanda, with indicators for information flow

5.6 Foreign Aid

Much of the HIS development and maintenance work was supported by foreign aid organizations. Organizations and programmes such as USAID[2], Jembi Health Systems[3], and HISP were present and aiding the country. When it came to outside funding, USAID stated that "Over the last 10 years, annual funding to USAID/Rwanda has increased from about $48 million in 2004 to over $150 million in 2012."[44]. Although outside funding had been a big supporter of the HIS in Rwanda, it had according to the World Bank recently begun to decrease[3]. This meant that the country had to become more self-reliant, and was partially the reason why the MoH had begun to look at computer systems and solutions that were cheaper than the ones that worked when foreign aid were paying the bills. A concern with losing funding was the potential loss of local staff that was originally being paid by foreign aid, but might lose their positions due to lack of funds. Some of these people like the HMIS administrators working at the MoH had become experts in computer maintenance and were largely carrying the projects. Without these key actors it would be challenging to keep running what was now working solutions that the HIS were based upon. Being able to reduce costs in other areas such as software licenses was then important to be able to maintain the human actors who largely supported the systems.

[2]http://www.usaid.gov/
[3]http://www.jembi.org/

Foreign aid was however still largely present. An example of a project that was supported by outside actors is the RHEA project. The Ministry of Health were working on a project which they called the Rwanda Health Enterprise Architecture in collaboration with organizations such as the Jembi Health Systems and USAID. The goal was to strengthen HIS and eHealth in the country by creating a national enterprise architecture. The plan was to use a set of national registers and computer systems for maintaining data on patients, facilities, and providers and link them all together and allow for easy information exchange between them[24]. They would link all these systems together using an interoperability layer which could make relevant data such as a national ID linked with a patient record. Figure 5.7 shows the envisioned structure in the eyes of the MoH, which includes some of the systems I have presented, in addition to a few others. The project primarily focused on maternal and child health in their pilot. It is interesting to look at this project as a sort of big picture view of the whole country context of the Rwanda HIS, where my participation can be seen as work related to the implementation of interoperability between underlying systems.

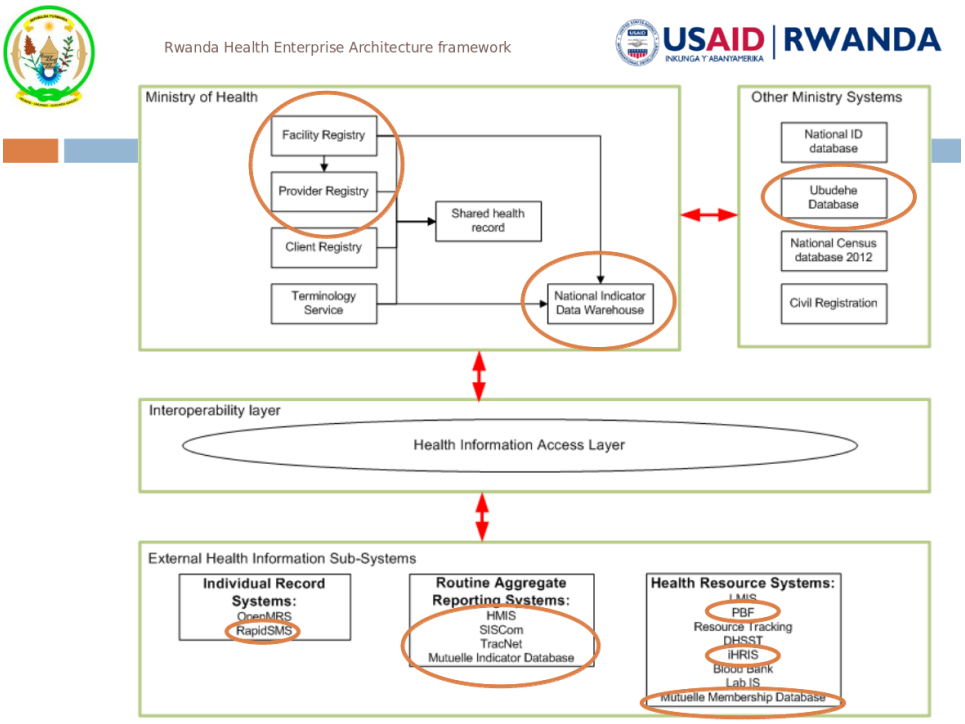


Figure 5.7: Overview of the MoH's plan for the RHEA.

A point that strengthens the hypothesis about independent projects by different organizations which can lead to fragmentations is how different actors look differently at the same project. The vision of Rwanda Health Enterprise Architecture in the eyes of Jembi Health Systems which was closely working in the RHEA project together with the MoH can be seen

in figure 5.8. What's interesting to note about Jembi's view of RHEA is that it is completely missing the DHIS 2 based systems, although they were the biggest computer systems currently deployed in Rwanda's HIS. Nevertheless, their views are quite similar and both show a vision of an interlinked architecture of the whole HIS of Rwanda.

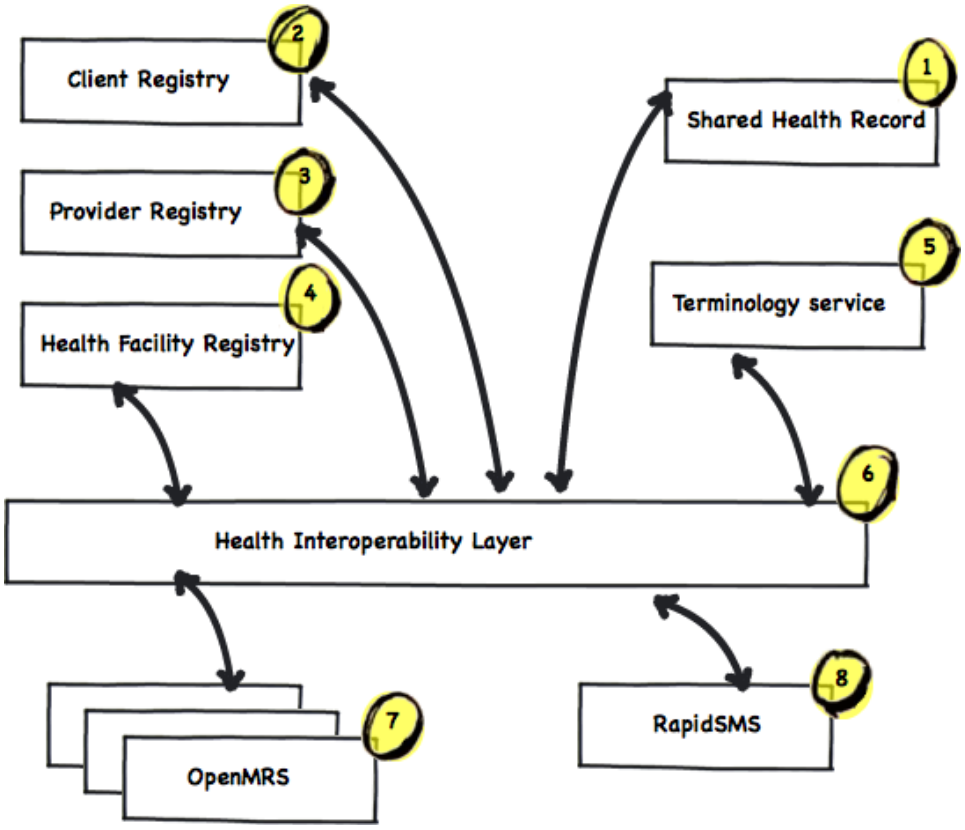


Figure 5.8: Jembi's view of the RHEA[24].

5.7 Development work

In this section I explain some of the work which I participated in that was being done to resolve the challenges with the HIS in Rwanda. The biggest projects were shifting from proprietary software to Open Source software, implementing SMS support with DHIS 2, and connecting computer systems.

5.7.1 Shifting from proprietary systems to Open Source systems

At the time of the visit there were still proprietary systems being used. Most of these systems were in the process of being phased out and replaced by free and Open Source software solutions. One of the major proprietary players was the TracNet system developed by the company Voxiva. I read

in a Master's thesis conducting field work in Rwanda a year before me[27] that it had been a lot bigger before, and had shrunk to only focus on tracking HIV patients. One of the main features that was being used in TracNet was the interactive voice response. Here community health workers would call in and report their reports by simple voice commands. The MoH wanted to replace the last bit of TracNet however. Since they had to pay for the minutes, they wanted to replace the IVR with an Open Source SMS based system. The IVR system was well functioning because many people were illiterate. Some members of the HMIS team did however think that it was probably possible to teach them how to enter simple data using SMS.

The two most obvious choices would be to either do it with DHIS 2, or to do it with RapidSMS. Since DHIS 2 was already quite big in Rwanda it was the preferred solution. It would allow them to use more of one system, which was good because they thought it was less complicated to have everything in one place. The previously mentioned master's student from the University of Oslo who worked in Rwanda a year before me developed a module for implementing SMPP support for DHIS 2 during his stay. According to other DHIS 2 developers it wasn't tested so it was unsure if it was functioning properly, but fixing it wouldn't be a big problem.

To set it up we also needed to get a short code subscription setup from a telecom operator in the country. A short code subscription is for having a short telephone number that is typically easy to remember, which is common when large organizations have SMS or phone-based services. Acquiring the subscription proved to take some time, and although the MoH already had short codes set up in other programs that they weren't using, it was for some reason hard to get. The projects in posession of the subscriptions were in the process of being shut down, so some thought the problems were in bureaucracy.

5.7.2 Implementing SMS support with DHIS 2

Two of the three tasks mentioned before which I worked on with the MoH in Rwanda were related to SMS reporting in DHIS 2. The malaria surveilance system was a new project in which they wanted to investigate the circumstances surrounding malaria cases, in addition to routinely sending climate and weather data from selected Health Facilities with the right equipment to monitor conditions. If they reported weather conditions that would be ideal for potentially malaria-carrying mosquitoes, they could then respond immediately.

The implementation of TracNet into DHIS 2 also needed the support of SMS reporting. TracNet was implemented using an Interactive Voice Response where health workers would call in to an automatic server, where a robot voice would take their voice input to report their statistical data. Since DHIS 2 didn't support IVR, it was decided that using SMS reporting was a suiting replacement.

The problem however was that setting up the SMS functionality proved to be a troublesome process. The proposed solution was to use the SMPP protocol, which is created to be able to handle large loads of SMS

messages. Figure 5.9 shows how the SMS messages are routed from when the community health workers send the SMS, until it ends up in the DHIS 2 database.

To set this up we needed to register a subscription with one of the big telecom companies. This was easier said than done, and for some reason we couldn't simply call and have it set up as one might think. The process of trying to acquire it was started in October, and was finally done in the middle of April, which can be seen as a long time to simply set up a subscription. The technology was definitely there, and the MoH had already gotten subscriptions for SMPP for the use of other services, so the reasons were not technological. For testing purposes of the implemented functionality in DHIS 2 there was also an attempt at getting a SMPP subscription from the telecom company Telenor in Oslo, which also ended in a road block. The supposed reason was that the SMPP protocol was very insecure and would grant access to potentially attack the Telenor server through the SMPP tunnel.

Another solution we attempted in regards to implementing SMPP support was to use a third party provider that offered an alternative way of getting SMPP support, but we were unable to make it work properly.

Regardless of the reason the setup of the short codes was slow, the plan was still to go ahead, and we could at least do preparations and set up as much as possible beforehand. Basically, we could do all the setup work with the DHIS 2 database and reporting forms, and have it up and running immediately after finally getting the subscription. Agreements had to be made as to how the data would be reported, and design the forms with relevant indicators. We could design the reporting forms before linking with the SMPP service, and we could test using other SMS solutions which did the same job, except that they couldn't handle large loads of SMS messages. For testing purposes that was no problem however since the number of messages would be minimal.

The way the SMS-reporting would work is that we would first register a set of indicators in a DHIS 2 reporting forms. The form of the malaria climate surveilance would take humidity readings three times a day, minimum and maximum temperature, and their averages. To link it with an SMS message, we assigned a code letter to each data element which we could use to link the SMS messages with the malaria surveilance reporting form. An SMS message would then typically look something like: "a70b80c75d25e31". See table 5.1 for an explanation of the elements.

Code letter	Value	Data Element
a	70	Humidity 1
b	80	Humidity 2
c	75	Humidity 3
d	25	Temperature Max
e	31	Temperature Min

Table 5.1: Description of elements in SMS message for reporting malaria surveilance

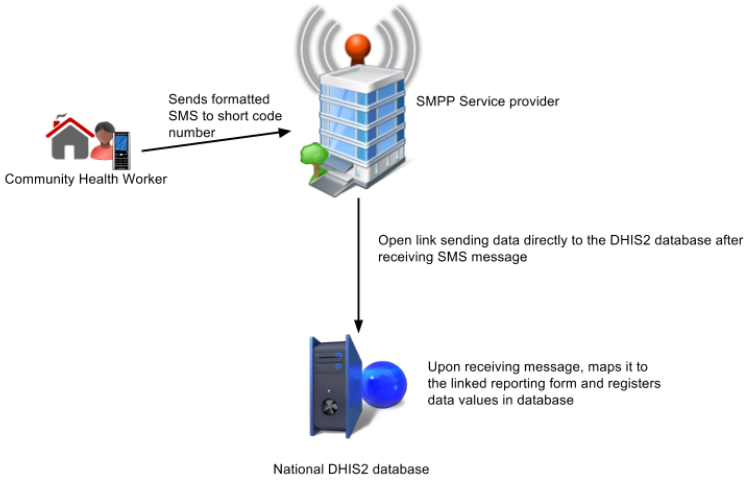


Figure 5.9: Flow chart of how SMS messages are linked into DHIS 2.

A further challenge with not having a working SMPP setup was that we were unable to test the implemented functionality with DHIS 2. SMPP support had been implemented a year before I went to Rwanda, but as mentioned it had yet to be tested properly. In theory it should've worked, but you never know with these things. The implementation was however simply using an Open Source library which worked in theory, so the chances were it would work out of the box.

When it came to implementing the TracNet support into DHIS 2, the technical challenges were the same as with the malaria surveilance, but the reporting form design and implementation with health workers tasks were given to the HMIS to do, as they already had substantial experience with creating forms. A supposed problem was that many of the health workers were illiterate, making it hard for them to switch from a IVR solution, to a text-based SMS solution, but again the HMIS staff were confident they could train them to send simple messages.

One of the upsides of having implemented SMS support in DHIS 2 was that it would open up for an array of possibilities to be built on SMS reporting in the future.

Chapter 6

Developing the Android application for DHIS 2 data analysis

In this chapter I describe the process of developing the Android application for DHIS 2 data analysis. As mentioned previously, one of the projects I worked on during my thesis work was developing an Android application supporting analysis functionality in DHIS 2. The app started out as a group project in a course at the University of Oslo run by HISP, after which it was further developed by myself over the course of about one year. The goal of the project was to provide an Android application which would make it easier for DHIS 2 users in analytical positions to easier be able to access data without having to use a web browser. The development process of the app was supported by the theory of Participatory Design as I worked closely with some of the intended users during the development stages. This was done through direct feedback, rapid prototyping, and testing of the app with the users personally, and by working in the HIS context of Rwanda as a whole to understand the day to day working situations where the application would potentially be used.

Figure 6.1: The Android application showing a full sized view of a Dashboard item next to the web browser Dashboard module.

6.1 What is an Android application

Android is the one of the popular operating systems running on smart phones and tablets[1]. It is Linux-based and created by Google. The Android operating system allows users to run programs known as “apps” on their handheld devices, just as applications are run on regular computers, except that they are specialized for a handheld touch device context. These apps are developed using the Java-based Android SDK which makes it easy to take advantage of the hardware of a given device, such as GPS, Internet connection, and motion sensors, in addition to making graphical user interfaces specifically designed for handheld touch devices. Developers can easily upload and make their applications available to the public through a portal called the Google Play, where users can download applications directly to their devices[29, Chapter 1].

6.2 Motivation for making the application

The main DHIS 2 system is web-based, and users need to use a web browser to interact with it. To reach the data you want inside DHIS 2 you then need to go through the steps of opening the web browser, entering the URL of your DHIS 2 system, and then log in. It may not seem like a lot of effort, but it does require you to be in front of a computer, and depending on the situation it could potentially prevent use if time is limited. There are also browser-based mobile versions of DHIS 2 where you don’t need a

[1] http://developer.android.com/about

computer, but you have to go through the same steps as just mentioned, and in a mobile context it can be a little more tedious when operating on a tiny screen.

The developed Android application reduced the number of steps to one, where that step was tapping your finger on an icon on a touch screen. Imagine being in a meeting where you want to show a diagram of statistics regarding cases of Malaria in a given country. Having to boot up a computer and then go through the previously mentioned steps, and then finding your desired data within the system is a lot more work than taking your phone out of your pocket, tapping on an icon and then selecting your diagram from your list of favorites. That was the goal of the application, to enable easier data access, which could increase data use.

For comparison figure 6.1 shows the web browser Dashboard and the finished Android application with a full screen selected diagram. The two devices are connected to the same live database located in Rwanda.

The functionality of the app supports some of the main modules of the DHIS 2 software, specially tailored for a mobile device context. It implements the dashboard, messaging, and interpretations modules, which are some of the main modules for data analysis in DHIS 2, and the app runs natively on any Android device. It also allows for multiple user accounts, and simple switching between them for users who need to connect to different DHIS 2 instances. The focus on implementing these three modules was inspired by a request from the main DHIS 2 developers in Oslo when the app was initially started. There was already an Android application for DHIS 2 which enabled data entry being developed at the time, but it had no functionality for viewing and analyzing data. Further it was thought that the DHIS 2 users who would have an Android device would be in higher up positions and using DHIS 2 for analysis purposes, which made it more interesting to make the app as there was a clear target user group.

During the course a proof of concept was developed, but it was far from usable for a normal DHIS 2 user.

Among the reasons for picking the app back up after the original course where it started finished and making a full fledged working version was a request from DHIS 2 users in Zambia. Some of their users were chiefs who were not very good with technology, and needed something simpler than the web browser to access their data. It was a perfect use case for the vision of the application.

Another reason was that I was going to Rwanda to work at the Ministry of Health together with people who fit the profile for the target users. I could then through the methodology of Participatory Design have a chance to develop the application to better suit the needs of the end users.

6.3 Functionality

The application supports three modules from the main DHIS 2 system. That is the dashboard, messaging, and the interpretations modules. Here

they are described in more detail.

The Dashboard

The dashboard module serves as a users front page. This is the first item encountered by a user when entering the DHIS 2 platform, both in the web interface, and in the mobile application. A "dashboard" consists of user defined sets of various graphs and presentations of data. These data elements can be either pivot tables, different kinds of charts (pie charts, line charts, etc.), and maps with indicators and legends. The different dashboards are available between users connecting to the same DHIS 2 instance, so that it is possible to share across accounts. The pivot tables and charts are also user defined, based on aggregate data collected throughout the country of a specific user. There was no ambition of implementing a full fledged DHIS 2 interface with all the functionality of the web interface, so users would still have to create these graphical presentations using the existing online tools through the web portal. An example of a chart could be the number of malaria cases reporting in a given country over a given period of time. Figure 6.2 shows a screenshot of the Dashboard module, and figure 6.3 shows a full size horizontal view of a single dashboard element.

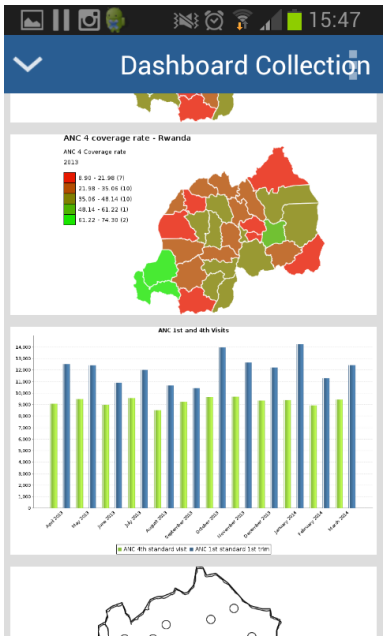


Figure 6.2: Screenshot of the Android implementation of the Dashboard module.

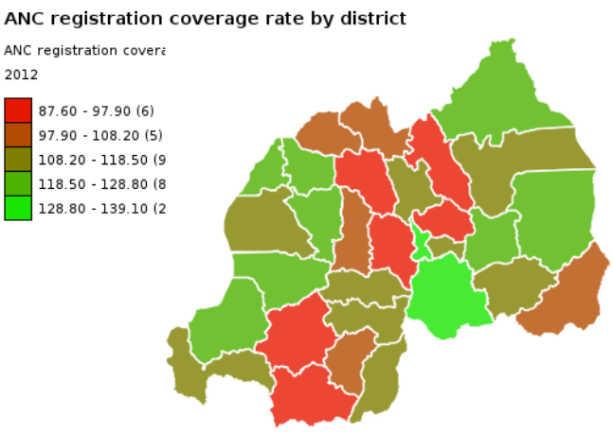


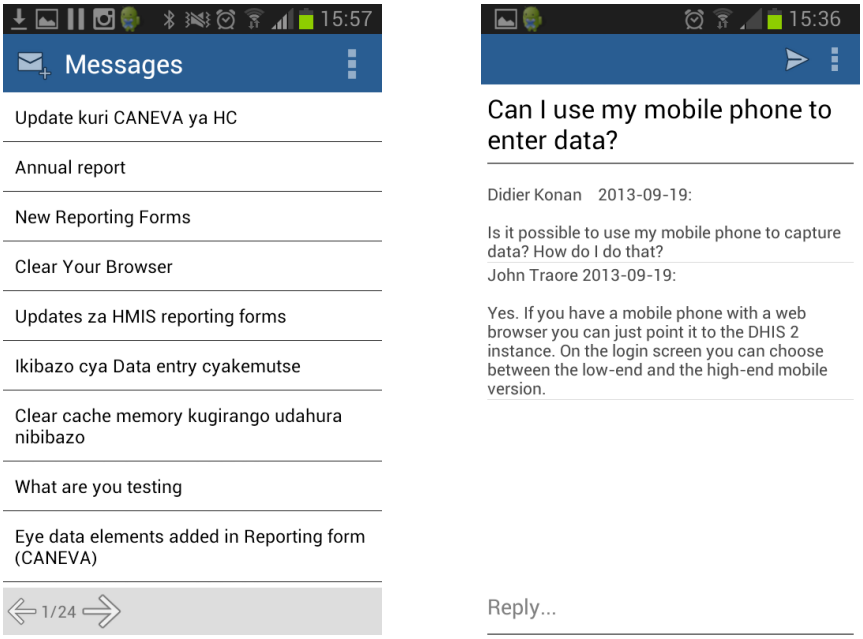
Figure 6.3: Screenshot of the Android implementation of the Dashboard module with a full screen horizontal view of one element.

Messaging

The messaging module provides a simple way for users to communicate with each other once they log in to the DHIS 2 platform. Users are able to have one to one conversations, or larger group conversations. It is also possible to send out messages to a whole user group, for example everyone working at a specific hospital, everyone in a certain district, or even a whole country. This provides an effective and quick way for users to exchange information, have a conversation or provide feedback. The aim was to implement the messaging module with the same functionality as the already existing one, but with a touch screen environment in mind. Figure 6.4 shows screenshots of the messaging module.

Interpretation Module

The interpretation module can be considered as a mix of the dashboard and the messaging modules. Here, users can discuss presentations of data, like the ones shown in the dashboard, to get a better understanding of why the data is presented in that specific way. A user will typically share an existing graph, and post a comment most likely in the form of a question. Then others will see these interpretations and comment back with their own thoughts. In this version the goal was to implement the functionality of commenting on existing interpretations. Functionality for creating and sharing new interpretations has not yet been implemented. Figure 6.5 shows a screenshot of the interpretations module.



(a) (b)

Figure 6.4: Screenshots of the messaging module. a) shows a list of conversations, and b) shows a single conversation.

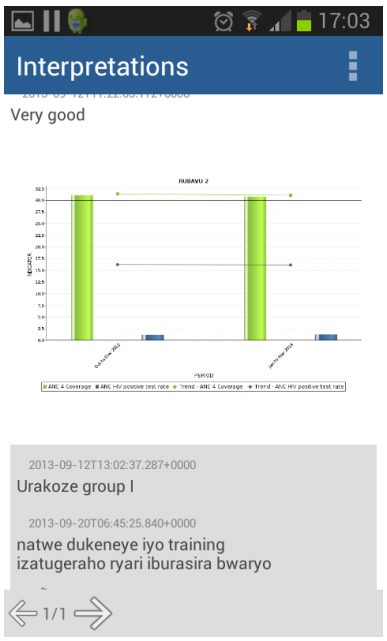


Figure 6.5: Screenshot of the Android implementation of the Interpretations module.

6.4 Target users

DHIS 2 is usually deployed in countries where financial resources are limited. Therefore the need for an Android app will not be present in many contexts, because there would be a lack of eligible users. There are however some exceptions, specifically in data analysis contexts. Therefore the target users for such an analysis application is applicable for are those in the higher up positions, rather than those working in the data collection part of a DHIS 2 based system. Here some roles are presented as examples.

Data managers and Monitoring and Evaluation officers

Many Health Center and District Hospitals have one dedicated data manager who is responsible for entering data into DHIS 2, and for providing reports for analysis. There is usually also one person responsible for monitoring and evaluation. Since there are not that many of these people compared to data collectors in each country, and they are in relatively high-up positions, they could more likely afford to have an Android phone.

Government officials

Government officials are usually of a higher income than many other DHIS 2 users, and will have the available funds to purchase an Android device. The application will allow the Minister of Health and other officials to access analytical data in a convenient matter on their device.

Less tech-savvy chiefs

Although not entirely relevant to the context of Rwanda, it is worth mentioning that a request was made from Zambia about a plan to give tablets to chiefdoms. The request stated that the chiefs are not all tech-savvy, and even having to log in through a web browser might prevent system use. Because of that they wanted a DHIS 2-app that automatically displays current data without the need of any technical knowledge. This request was one of the main points of motivation for developing a working version of the application.

6.5 The development process

The app started out as a collaboration project in a course at the University of Oslo where it was developed to function as a proof of concept. The functionality that was explained in the previous sections was suggested by the main DHIS 2 developers at the University of Oslo when the idea of the app was being developed. Since Android phones can be seen as costly in some contexts it was targeted towards those in higher positions who would analyze data.

It was decided that developing the application using the native development kit was the way to go as it offers the best performance. It is possible to develop using frameworks that allow you to automatically make the application for more platforms, but it has performance costs, and we wanted the app to be as effective as possible, especially considering offline availability and moderate internet connections.

After the course was done all the basic functionality was there, but it was far from ready for deployment to end users. With the interests from users and the fact that I was going to Rwanda to work at the Ministry of Health I picked the project back up and worked towards developing a functioning version that could be released to the public.

When I was working in Kigali I could use the theory of Participatory Design while closely work with the intended users to better address the desires of the people the app was developed for. Typically I would follow an informal cyclical process where I would show the application to the DHIS 2 involved staff who could give me feedback on the layout and functionality. I could then make changes to get new opinions based on the improvements, and make changes again. Some of the functionality came to be because they asked for it, one of which being the ability to enable a full screen zoomable view of Dashboard elements such as graphs and charts. The use case of that particular feature would be to not have to print charts on paper, but rather to be able to show it on a phone or a tablet whilst in a meeting or some other situation.

The application wasn't released to the Google Play portal[2] until I returned from the trip and was uploaded on March 1st 2014, but testing on live devices was still possible by installing an application package file directly onto the devices.

6.5.1 Connecting with the WebAPI

The application is meant to work together with an existingly set up DHIS 2 instance. The data that is used and displayed in the app is fetched from the database that stores data reported by health workers within a country context. To get the data, a connection is made to the WebAPI of DHIS 2, using a URL, a username, and a password - the same password a normal DHIS 2 user would use for any other purpose with the system. The way the WebAPI works is by enabling users to navigate through data using different URLs. Depending on the entered URL, the system returns JSON formatted data which can be either raw data, images, or more data making it possible to navigate further. Figure 6.6 shows an example of a JSON output of a Dashboard containing two elements, a map and a chart. To navigate further, for example to see the image of the chart, you would use the id of the chart as a part of the URL in the next request. Note that the current URL in the top of figure 6.6 is containing the id of the Dashboard itself.

The WebAPI also has the support to allow for data input to the server, for example for sending messages. This is used in the modules for

[2]https://play.google.com/store

messaging and for interpretations.

Figure 6.6: Example of JSON output from the WebAPI which shows a list of 2 elements in a Dashboard called “Immunization”. The elements are one chart and one map.

6.5.2 Affecting the DHIS 2 back end through Participatory Design

While I was developing the application I often found that the WebAPI lacked some of the support I needed for the app to work. To solve this I could easily ask the DHIS 2 developers if they could add it by e-mailing them, which they often did right away. Since I was testing the application directly to the development server the changes would be ready for use within a few hours. The following of the Participatory Design principles was

the back bone for finding the need for additions to the WebAPI. Some of the suggested functionality was impossible to implement due to limitiations in the WebAPI itself, but by communicating with both the end users and the back end developers the problems were solved. In this context I can be seen as an intermediary translating the needs of the end users to the back end developers of the core of DHIS 2. By communicating with the users personally and by getting to know their working situations I could identify the missing support in the back end which would enable for the creation of what would satisfy the desired functionality. The added functionality in the back end would also be available for everyone else using DHIS 2.

6.6 Usage statistics and feedback

As mentioned, while working out of the office at the MoH in Rwanda I would test the application with the key DHIS 2 personnel during the development process, who provided their feedback. When they had questions about functionality that wasn't working properly or differently from what they wanted, I could change the code, and come back with an updated version the next day.

In addition to getting feedback from testers while developing the app, there was feedback after the first version of the app was finished and released. After one month of being available to the public the app had been downloaded in more than 10 countries, with the top countries being Ghana, Norway, Zambia, DRC, India, Kenya, Myanmar, and Rwanda. Using the Google Play portal for uploading the application not only makes the app available for everyone with an Android device, it also gives feedback to the developer by providing statistics based on user data. These statistics make it possible to see the number of downloads and active users on a country basis, and is updated daily. Figure 6.7 shows the current statistics as of May 17th 2014, filtered by country. It is possible to filter based on other criteria as well, for example by device, but in this context it is most relevant to show by country. What's interesting to note is that Ghana by far has the biggest user base, even though none of the actors in the country were involved during the development process. The reason for the high number of downloads in Ghana has not yet been investigated.

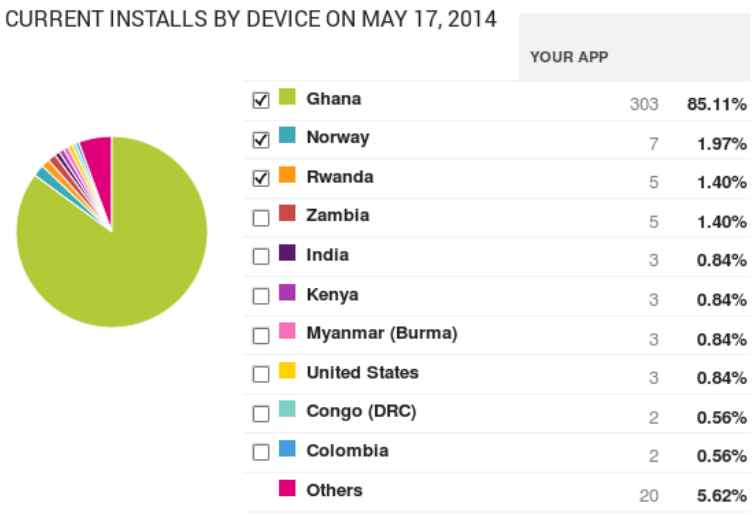


Figure 6.7: The Google Play Developer Console displaying stats of the number of downloads filtered by country for the application.

The feedback after the final version was released was generally good, and it indicated that there were definitely usage areas for the application. The System Administrator for all DHIS 2 use at the Ministry of Health in Rwanda stated that after he had tested the application on his own, he wanted to install it for other users in an email conversation saying "My intention is to install this app into the Hon. Ministers apps, senior managers apps for quick image. We are happy to be involved in its improvement. ". He was one of the people who I was most involved with while working at the Ministry of Health in general, and during the Participatory Design process of the application. In figure 6.8 he is showing the application installed on his own phone, while at a training workshop for DHIS 2. Later, when the same person went to a workshop in Zambia he installed the application on the team's phones and provided feedback from them. Some of the feedback was bug related and for requesting new features.

The app was also announced in the DHIS 2 mailing lists. The feedback from the users was highly general, but it too indicated that there were definitely users interested in using it. Some of the responses were related to support of the application for some users who had problems with understanding how to connect to their respective database. Others simply wanted to give thanks.

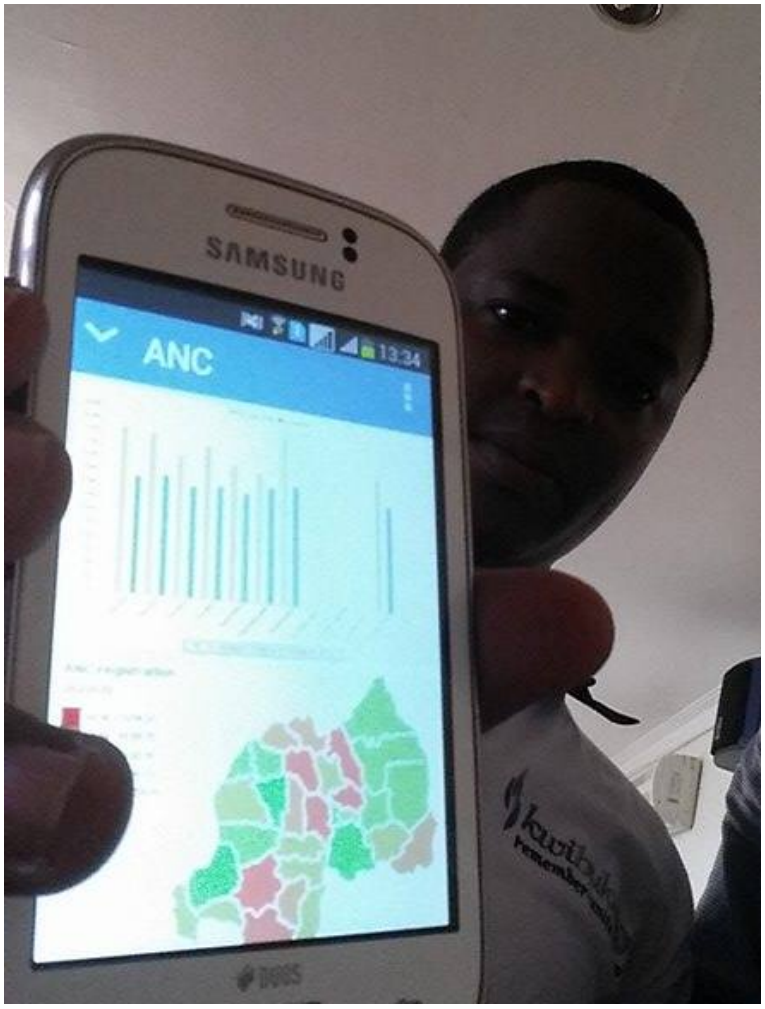


Figure 6.8: The DHIS 2 System Administrator in Rwanda showing the application while at a training workshop.

6.7 Future Work

In addition to this Android app, there are at the time two other Android applications related to DHIS 2. One of them is an application that transforms the phone into an SMS modem, the other one is a data entry application. Since all three of the apps are DHIS 2 related, it could be interesting to merge them into one application. If a user wanted to use the functionality of all three, it would be easier to not have to switch applications to switch between tasks. That is especially true for the analysis app and the data entry app as they both handle DHIS 2 data. They further both connect to an existing DHIS 2 instace, which probably would be the same database for both apps for a given user. It would then be unecessary to have to log in to both apps, rather than in just one. Merging the two apps together would be trivial as they could still function as individual modules within a containing app, allowing for switching between them.

Furthermore, there is still testing and possibly bug fixes that need to be attended to, and additional functionality can be added and improved. The application is definitely working, but there is still room for improvements, and the fact that it is released as Open Source will hopefully encourage others to contribute as well. The profile for the application in the Google Play portal provides a link to the online repository[3] where the source code can be downloaded so that it is easy to find if someone decides to make additions.

[3]https://code.launchpad.net/~simernes/+junk/dhis-android-reporting

Chapter 7

Discussion

In this chapter I discuss my findings and link them with the literature presented in the Literature Review, and the research topics. As a recap these were the topics presented:

- Find out how the HIS in Rwanda works and how it relates to development challenges such as capacity and sustainability.
- Investigate how Open Source software can be beneficial in the developing country HIS context with regards to lower costs and local capacity building.
- Look at challenges of fragmentation of computer systems and look at possible solutions.
- Find out if and how an Android application for analysis purposes in DHIS 2 is applicable and useful in the low resource country HIS context and see how the Participatory Design methodology can contribute to making the application better.

I wanted to find out how Open Source software could be beneficial in regards to lower costs and capacity building to improve sustainability in HIS. To do this, I wanted to understand the workings of the Rwandan HIS, with a special focus on the computer systems being used. In the HIS context of Rwanda I further wanted to look at problems with fragmentations of computer systems, and at ways of addressing such problems, and if Open Source software could also be benecifial in that area. Lastly I wanted to use the Participatory Design methodology as a tool while developing an Android Application for supporting DHIS 2. I wanted to use the method to better understand and meet the needs of the intended users, and I wanted to use the situation where I was working at the Ministry of Health as an opportunity to get to know the people who would participate in the Participatory Design.

First I discuss the solutions to fragmentations of computer systems that were currently taking place in Rwanda. These are developing an interoperability engine for automatically linking systems, and the other is reducing and merging systems. Next I talk about the process of

developing the Android application and how Participatory Design has positively affected the process. Then I comment on the HIS in Rwanda, and how it relates to sustainability and capacity building, before I discuss how Open Source software is beneficial for sustainability through lower costs, the ability to change, and through the encouragement of capacity building. Lastly I reflect on my approach to the conducted research, and summarize the discussion in regards to the presented research topics.

7.1 Solutions to the problems of fragmentation of computer systems

In this section I discuss two approaches that were made to attack the problems of fragmentation of computer systems in the Rwandan HIS. The first is making an application that automatically links systems together, here called an Interoperability Engine, and the other is reducing systems by merging one into another.

7.1.1 Developing an Interoperability Engine

While in Rwanda and after I returned to Norway I took part in developing an interoperability engine for connecting the main systems used in Rwanda's Health sector. The engine made use of endpoints from each of the systems such as data imports and exports and WebAPIs. More specifically, an interoperability engine here means an application that automatically connects different software systems together, with the intention of reducing manual work needed to perform actions across systems for synchronizations and other forms of cross-system communication.

Challenges of fragmentation of computer systems is not only present in Rwanda, and in [20] the author explain that organizations' data is often distributed across multiple heterogeneous information systems, and that exchanging data between them is seen as difficult. Looking at Rwanda's Health sector as a large organization, shows an example of this.

The case of having large and complex systems that grow to an unmanageable state is a common case, and the authors of [48] describe and call such systems Legacy Information Systems, which can be defined as "any information system that significantly resists modification and evolution". Furthermore, one of the biggest challenges with such legacy systems they present is that "integration efforts are greatly hampered by the absence of clean interfaces".

There were several challenges with heterogeneity of computer systems, one of which being that in Rwanda they were using 4 instances of DHIS 2 instead of 1 as intended by the developers. This resulted in problems, especially with synchronization of the registered elements in the database that were relevant across the systems, which was one of the main reasons for making a interoperability engine. Making the engine was however somewhat problematic as the system really wasn't designed for such tasks.

Another motivation for making the interoperability engine was that they were using a wide range of other computer based HIS, where data was relevant across them, in addition to being relevant to the DHIS 2 based systems. Like with the DHIS 2 systems, there was a desire to automatically link the systems so that some relevant data could be automatically or at least semi-automatically available between them. Mainly the desire was to connect the non-DHIS 2 systems with the DHIS 2 based systems as DHIS 2 worked as a data warehouse, but seperating the tasks into two made it possible to look at the DHIS 2 based systems as one entity, after they had been linked together in the first place.

The need for operating across systems and databases was not a new problem in Rwanda, and was already often performed by manually running SQL queries on the back end databases. Since that was not a part of the intended use of either system, this could become overly complicated, and significant knowledge of the SQL language was required. If not done right it could also lead to problems with the rest of the functionality of the different systems.

Implementing solutions for interoperability between computer systems is neither a new topic in general. The Chicken Little strategy proposed by Brodie and Stonebreaker[11] offers a guideline of 11 steps of which to be followed to achieve migration, where they use gateways as a means of communication between systems. The ultimate goal here however is to completely migrate from one system to another, rather than creating a persistent and automatic link, which was the intention of most of the interoperability work being done at the MoH in Rwanda.

The engine in our case was developed as a set of specifically defined routes which automatically went through different sets of scripts which would pull data from the one end, change the format from the origin system to a format which could fit into the destination, and then pushed into the receiving system. Such means of connecting systems are described as gateways in [17], and figure 7.2 shows the basic flow of how such routes work. Each route had to be specifically tailored for each conversion.

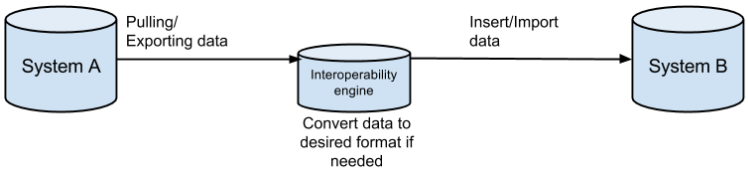


Figure 7.1: Flow chart of a single generic gateway.

After having identified the problem and the technical specifications of what needed to be synchronizied, work was started on developing the application to automate the process. While implementing it we quickly learned that some experimentation work had already been started by an involved Ph. D. student based in Dublin who had long been working on the HIS in Rwanda. It was not necessary to develop both applications, so it was

better to join in and help on his work, rather than fragmenting the HIS even more. At first there was an attempt to implement a solution using the Open Source Apache Camel framework[1], but it proved to be unfit for the intended use. In the end it was implemented using a set of PHP scripts with the help of some SQL scripts and simple JSON, and XSL scripts for formatting.

Connecting the DHIS 2 Instances

DHIS 2 is meant to be run as one single instance for a whole country. This means that when you use four instances instead like they were in Rwanda, there will be some implications. The biggest challenge relates to data integrity and synchronization, as the databases are working independently of each other. A DHIS 2 database typically consists of a set of data elements, indicators, users, and organization units in addition to other elements. All of these objects are entered by the users into the system's database. This means that if you enter something in one database, it will of course not appear in another. In our case it means that you will have to create the same objects four times, to have the databases fully synchronized. Since it is possible to alter the objects after they are created, the integrity of synchronization of possible changes is also something that needed to be taken into consideration.

Because one of the DHIS 2 instances functioned as a Data warehouse, taking selected data from each of the other instances each month, there was also a need to export data into it from the others. At the time of visiting, this was done manually by performing SQL queries on the underlying databases in the back end of the DHIS 2 systems. It was a somewhat tedious process, although it worked, but it did require the person transferring the data to have significant skills in the use of the SQL query language. There was also at the time functionality making it possible to export and import data, but it was limited and did not suit the needs of this particular case. There was a function allowing you to export metadata using the user interface, but it did require you to export all of the data, instead of just selected elements. Because of the size of the databases this was not a possible solution. Another function existed using a WebAPI which enabled the user to export data values, which means the collected aggregate data. It was however not possible to select a list of data values to be exported, which meant that you would have to create a sort of manually created request selecting all the desired items. In our case this meant a list of about 80 elements, making the work quite tedious, not to mention making the request long in character size. The WebAPI also worked for exporting metadata, but was again limited to exporting all or nothing. Exporting everything was not an option however, since that would flood the different DHIS 2 instances with each other's data elements and such.

Although there were challenges using the existing functionality we could request features for coming updates of DHIS 2 quite easily. We were able to communicate with the developers informally through e-mail and the

[1] http://camel.apache.org/

needed functionality would arrive later in coming versions of DHIS 2.

Until the updates arrived however, only a few of the desired interoperability problems could be solved, but we could at least identify and plan for the future. We worked on solutions together with the Ph. D. student in Dublin, and managed to create a series of PHP scripts which used an XML-based metadata export function of DHIS 2 that could synchronize the organization units automatically. This script would run automatically every 10 minutes or so, so that whenever someone created an organization unit into the HMIS database, it would within 10 minutes be duplicated into the 3 other instances (figure 7.2). It would also update organization units that had been made changes to. The only missing aspect was that it did not take into account deleted organization units, so that would have to be done manually for the time being.

The first time the synchronization job was run, more than a hundred organization units were introduced into the other systems. In the future, new ones would automatically be created in each system whenever one was made in the HMIS.

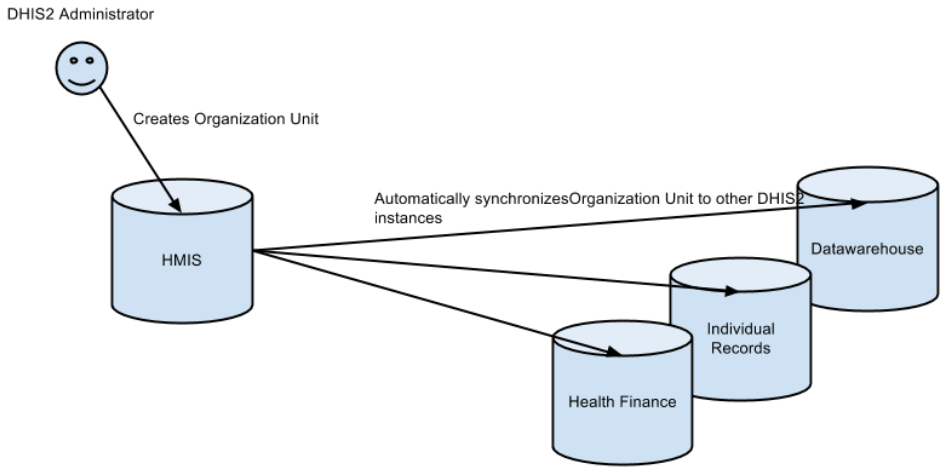


Figure 7.2: Data flow indication of organization unit synchronization in DHIS 2 systems

Connecting the other systems

The data warehouse instance of the DHIS 2 based systems did not only take data from other DHIS 2 instances, but also from other systems, and since none of the systems followed a shared data structure standard, there were challenges around moving data from one to the other. One way to solve such problems was to make use of exporting and importing functionality in both ends, and performing hard coded manipulations in between. That means that data was exported to an xml file from one of the systems, then the interoperability engine would change the data into a format readable for DHIS 2, and then import it into the DHIS 2 data warehouse.

One such route was developed for importing data from the TracNet system into DHIS 2. The DHIS 2 administrators could now easily upload a TracNet-exported data file into the interoperability engine's simple web interface and trigger a data import into DHIS 2. There were some questions around how relevant the TracNet route actually was since the TracNet system was in the process of being phased into DHIS 2 anyways, but it did not take a lot of time to make so it didn't matter much.

Another interoperability route that was developed was for exporting organization units from DHIS 2 into the Health Facility Register. Creating this route was relatively simple as all the needed functionality was there both on the DHIS 2 side, and on the HFR side. Now, whenever an organization unit was entered into DHIS 2, it would also be pushed to the HFR, which then could be accessed country-wide for an up to date, synchronized list of organization units. This list was both human readable in addition to the WebAPI that enabled computer automation.

Work was also done on trying to figure out how to import data from other systems like OpenMRS into DHIS 2. It was however challenging since the functionality for exporting data in OpenMRS was quite limited at the time. The provided XML report was hard to work with and lacking in structure, which made it challenging to extract the relevant data needed to export into DHIS 2. Since there was a team of OpenMRS developers working at the MoH we could provide feedback in open discussions to move towards a working solution.

Comments on the Interoperability development approach

The linking of computer systems and interoperability between them was generally in Rwanda handled in three ways. These are 1) connection through interoperability engines using gateways, 2) reducing systems by implementing functions of one system into another, and 3) manually exporting and importing using SQL queries and data export and import functionality.

The method of using an interoperability engine to connect systems through the use of gateways as described in [17] and [5] is an approach that has yielded immediate results without requiring much work in the presented cases. It allowed for synchronization between systems and reduction of work in maintaining data integrity on the higher level by automatically synching entered data from one system to others.

A downside to this is that by developing an external application like the interoperability engine, you are adding to the complexity of the HIS by increasing the number of systems.

In some of the cases it was possible to develop gateways (or "routes") using existing export and import functionality and WebAPIs. There were other cases where more hard-coded solutions, for example using SQL queries directly accessing the underlying databases were used as a part of the gateway compositions. The reason why such hard-coded measures had to be made was because the functionality of the systems didn't allow for the task to be performed using them, and the SQL query approach was the

only way to immediately solve the problem. This is similar to the problem posed by the authors of [5] who describe one of the main problems of larger information systems of being that "integration efforts are greatly hampered by the absence of clean interfaces".

Although the SQL query based gateways performed their duty at the time, it may not be a sustainable solution in the long run. This is because the queries have to be specifically designed for each single task and can become unmanageable and hard to maintain if they are very complex and specific. A problem can also be that if you keep following this approach, you will end up with a lot of different specific SQL queries on top of the different specific gateways. It could also be a problem if the underlying database structure changes over time as the software is developed, causing the queries to potentially break. In addition you are dependent on having competence in SQL and knowledge of the back-end databases of different systems. A further implication and challenge of creating such SQL queries at least in the case of DHIS 2 is that because the database structure is quite complex, it is challenging to make sure that you are not making mistakes which negatively affect the system by bypassing the normal data input procedures. There are cases where tables and values are being generated in the back-end system based on inputs which can some times be hard to account for when making direct data inserts. A really unlucky outcome could be to potentially break the whole system if done incorrectly.

By using endpoints such as WebAPIs or import and export functionality to build gateways, the risk of such problems occuring is eliminated. Since it is a part of the functionality of the computer systems, the back-end database handling is accounted for, just as all other functionality of the systems. The downside of this approach is that you have to rely on what is supported by the endpoints. In the case of developing interoperability gateways in the Rwanda HIS, some of the tasks we wanted to perform were not supported, which is why it was decided to develop them using queries instead. During the process of development we did communicate closely with the DHIS 2 developers in Oslo, who were happy to accept new tasks, but could not help us solve the problem right there and then. Additionally some of the requested functionality was already on its way into DHIS 2, but we would have to wait for a new version to be released, and then installed in the country before it would be possible to make use of it.

Generally speaking, using functionality provided by the different systems is a safer approach, but when it's necessary it is possible to use queries to get immediate results even if it is more risky.

On the other hand, others have argued that using gateways is not necessarily a good approach. The authors of [49] argue that the use of gateways "adds greatly to the complexity of an already complex process and is also a considerable technical challenge in itself.". Furthermore, the authors of [4] present three negative points of such approaches in that they:

- "offer no support for transaction management and thus no way to ensure data consistency between the legacy and target systems;
- provide no way to homogenize the structural and representational

differences between the two database schemas; and

- are difficult to build and operate."

However, since the cases presented here were relatively minor it was an effective way to solve the problems at the time using gateways. The problems were also somewhat different in that those papers discuss interoperability with an ultimate goal of system migration rather than maintaining automatic links.

7.1.2 Reduction of systems

Another approach to solving the problem of fragmentation is the reduction of systems, and rather having bigger systems performing more tasks. This is perhaps more similar to the intended use of the presented methods in [49] and [4]. The Ministry of Health had started reducing the number of computer systems in their HIS. To do this, they were phasing out existing systems, and re-implementing the functionality in other existing systems with the right support. In [4] the authors describe this process as what they call Migration. Their definition of it is that it "moves an existing, operational system to a new platform, retaining the legacy system's functionality and causing as little disruption to the existing operational and business environment as possible.". Although it is described as a very complex procedure compared to other approaches, they argue that the long term benefits may weigh up for the additional efforts as it "offers more flexibility, better system understanding, easier maintenance, and reduced costs.". In this context what the authors call the legacy system are the systems that are being replaced and reimplemented in DHIS 2, which in their description would be the new platform.

At the time the MoH were phasing out two proprietary systems using mobile phone functionality, replacing them in DHIS 2. DHIS 2 could not support exactly the same functions as the proprietary systems were using, but was able to enable implementations of similar fashions, performing the same basic tasks. The main big change of technology was that the IVR functionality was to be replaced with SMS reporting. The approach they followed was similar to the Chicken Little approach presented by Brodie and Stonebreaker in [11] and used gateways in the form of important and exporting functionality of the two systems, in addition to some SQL database manipulation to transfer the data.

By doing this, they were handling the problems of duplication and system fragmentation. When conducting interviews with staff at the administrative levels of the HMIS it was generally seen as a positive thing if it was possible to move more and more functionality into one system, moving away from the fragmented way it was done using different systems for the different tasks.

The process was however lengthy in time, and was not completely finished while I was present, and may support the notion of the authors of [49] who view the use of gateways as a non optimal approach. In

[5] it is however stated that “Legacy information system migration is a long process, typically lasting five to ten years”, where the authors discuss gateway based approaches, and the Butterfly approach, but it does at least shine a positive light on this particular case, as the migration process has been in progress for less than two years and is closing in on the finish line.

Regional Developments

Another project worth mentioning is that of implementing a regional database collaboration project in the East African Community. The plan is to select indicators of health data to be reported to a shared database to strengthen the relationships and HIS, and to make it possible to compare health data across borders. The fragmentation of having many computer systems becomes more complex if you are to connect parts of the HIS of different countries who all consist of various different computer systems. Having less systems in this context could contribute to making it easier to gather data to a shared database. An interoperability engine and reduction of systems as previously discussed can make the process of cross-country HIS interlinkage more manageable for the same reasons as in interlinkage within a country HIS.

7.2 Developing the Android Application

During the thesis period a working Android Application was developed and released. One of the interesting aspects of developing the app was that I got to work on a self contained project within the DHIS 2 context. Developing for the core of DHIS 2 can be a challenging task as you have to go through the approval processes of the core developers for it to be accepted which is the common case of Open Source software projects in general [30]. In my case it was easier to make a larger impact by implementing a stand alone project on top of the existing DHIS 2 installed base. Working on a project independently of others is sometimes easier as you have free boundaries on how you work and won’t have to spend time on collaboration efforts. However, it can also be a negative aspect since you don’t get as much productive criticism as you would if you were working in a project team.

The project wasn’t completely self-contained however, as the use of the WebAPI had to be taken into account, and because I frequently got feedback from some of the core DHIS 2 developers at the University of Oslo who were helping guide me in the right direction towards the completion of the application. This feedback was more regarding the user interface, functionality, and front end part of the application which the end users would see, rather than the underlying technology used to achieve it. In addition, others were involved due to the use of the Participatory Design methodology as collaborators who took part in the development through feedback and user testing with the HMIS staff in Rwandas Ministry of Health.

That being said, as the project is released Open Source, other might

decide to contribute, which would make it into more of a collaboration effort.

On a sidenote while the positive aspects of self containment in a project is being discussed, having previously discussed how the reduction of systems in HIS can contribute to effectivity it may seem odd to develop an additional application like this Android application. It is possible to argue that making an application on top of another system may add to fragmentation and complexity, but since it is connecting directly to DHIS 2, it can be seen as a part of the software base that is DHIS 2, working as an extention rather than an additional system.

When it comes to the collaboration efforts in updating the WebAPI an interesting part of this process is that I was able to request new features to be added to the WebAPI on the go through contact with the other DHIS 2 developers. Whenever the WebAPI was unable to satisfy the requirements for some piece of functionality of the application I could talk to the developers personally, and the changes would arrive shortly after. I could've also implemented the changes myself, but in most of the cases they were done quickly by the experienced developers who immediately knew how and where to make the changes to solve the problem, and who had the power to push the changes to the online repository.

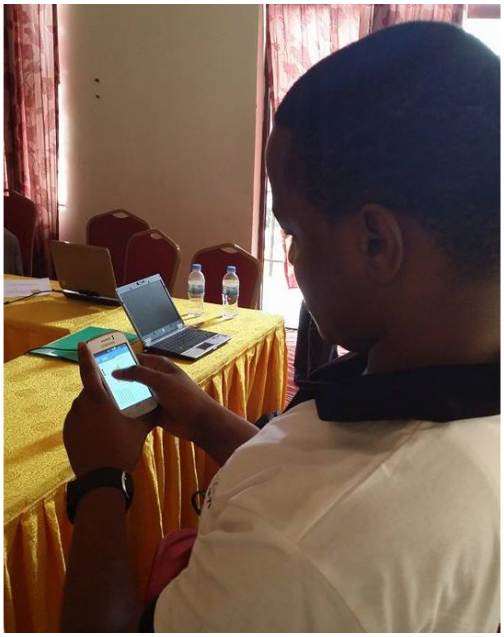


Figure 7.3: The DHIS 2 Database Administrator in Rwanda using the application at a workshop.

7.3 The Rwanda HIS and Sustainability

The HIS in Rwanda included a variety of computer based systems. The country had deployed functioning computer systems for the health sector

nation wide, and were developing their computer HIS further every day. Network coverage was country wide, allowing for Internet based systems in even the most remote regions.

Most systems performed their own specific task, and the people working with them usually only used one of the systems, especially if it was on the lower levels of the health sector. Most of the time the systems were self-contained, and performed its specific task without connecting to other systems. The systems were implemented using a variety of softwares, most of which were generic Open Source tools which allowed you to design and make use of the functionality you needed. The softwares were specialist in each their particular area like for example aggregate data collection and analysis, individual patient records, human resources, or registry functionality.

DHIS 2 was the main software being used for aggregate data collection and was also covering individual records in the field of tuberculosis treatment. Data managers in local areas entered data directly into DHIS 2, making the data immediately available for others to analyze. Health workers also used other computer systems in other programs, in which some of the data from the other systems was entered into the DHIS 2 Data warehouse which stored the most important statistical data for the country. The DHIS 2 Data warehouse can be seen as a central database for the whole country which could be used to generate forms and charts, which could then enable easier decision making based on live data.

Even though the systems were designed with particular tasks in mind, data was sometimes duplicated or relevant across systems and caused redundant work in synchronization efforts as it was explained in the previous chapters. When data was transferred from one system to the other it was often done manually. If these processes were automated it would save time and enable more work to be done more efficiently.

The MoH were working every day to improve their HIS. They implemented new reporting forms and new analysis charts based on decisions made by programs and managers, in addition to trying to automate processes that had to be manually performed.

Furthermore it is important not to forget that an important part of the HIS are the people working in it. The information itself is invaluable but if there were no one to use it, it would be useless. There were often training workshops where MoH staff taught health workers how to use the systems in their local areas. It varied from teaching new functionality introduced in systems for data managers who had already been using the software, to basic training of new health personnel.

7.3.1 Capacity building and sustainability

As it was mentioned in the introduction, a big question in the context of HIS in developing countries is the one of capacity and sustainability. At the moment, many of the projects rely on funding coming from foreign aid organizations who work with improving the HIS. However, an ultimate goal is that it will be self-sustainable in the future, without the help of

foreign aid[7, Introduction]. For that to be possible, the HIS needs to be run without the need of financial and other support from foreign agencies. There is however often a lack of resources and capable people able to maintain HIS efforts, which makes building of capacity with training and knowledge an important part of HIS development. The systems should preferably be run and maintained by local staff, which is explained by the authors in [23] where they argue that the success of a sustainable large scale HIS depends on it. At the moment, the majority of staff were in fact locals, but were often supported by foreign organizations.

The fact that the use of DHIS 2 is encouraging local capacity building is a move in the right direction. The authors of [6] explain how capacity is developed within the HISP network, where one particular area is "DHIS-related IT capacity". The staff that is being trained become valuable resources as the number of people capable of performing the required tasks are limited. Both in special training workshops, and by using the system they are becoming more skilled every day, and are able to train others using their own acquired knowledge. The main administrative HMIS staff participated in training sessions conducted by the HISP, where they learned how they could implement and improve the system in their own country, and train staff on their own. In [42] these training sessions for the administrative staff are presented, and called workshops, and are run under what is called the DHIS Academy.

In Rwanda they were also further improving their systems every day by implementing more features, reporting forms, while having their database grow through data collection. Since the local staff were maintaining and administrating their own servers and databases it means that they owned everything that they were reliant on, including the computers that hosted the systems. This is further contributing to their local capacity, rather than them being dependent on outside actors owning and maintaining their systems.

Training workshop in Kayonza

An example of such capacity building is a training workshop I attended, where data managers were trained by the main HMIS staff on the use of DHIS 2.

At the time of arriving the team responsible for DHIS 2 in Rwanda were conducting a training session in the Kayonza district in the Eastern Province. The training lasted for a week, and was about training community health workers and data managers at Health Facilities on the analytical functionality of DHIS 2. Rwanda had been using DHIS 2 and entering data for about two years when we arrived, and were now ready to start using the data through generating various types of graphs, reports and pivot tables. The training took part in a hotel, and they had invited all of the attendants (roughly 250 people) to stay there for the week. To conduct the training they had brought with them a computer with an installed instance of DHIS 2 acting as a server with a wireless router connected to it. This allowed everyone to connect and work without the need for an internet connection,

and without having to interfere with the live DHIS 2 instances running in the country. The hosts of the training were most of the administrators responsible for DHIS 2 working at the MoH.

Training workshops like this happened quite often, meaning that the capacity of available people capable of using DHIS 2 within the country grew rapidly. The staff were commonly local Rwandans who worked in their respective districts, effectively creating local capacity. During my stay of five weeks, two such week-long training sessions were held.

The capacity of users of DHIS 2 grew rapidly, but the fact that the main HMIS staff at the MoH frequently spent a whole week at local training sessions away from their actual MoH offices shows how important and willing these few key people were. What it also shows is how busy they were, and how diverse their work tasks were. The data managers and the other users would not be able to use their skills in the system were it not for the administrators, which means that they were all relying on these 8 people. Building capacity in the form of normal users such as data managers is important, but training of administrative staff and super users is perhaps even more important since one cannot exist without the other.

7.4 The effect of Open Source software on sustainability

One of the goals of this thesis was to find out how Open Source software could contribute to sustainability. Here I discuss two aspects where Open Source software is applicable. One of them is how it is possible to changes to systems where the source code is available to either implement better gateways for enabling interoperability or for implementing new features which could enable reductions of systems through migration. The other is how it reduces the costs of the computer systems, and how it encourages local capacity.

7.4.1 The ability to change Open Source software to tackle fragmentation

Two ways of handling problems with fragmentation was just discussed. The use of gateways for enabling system interoperability, and system migration. However, sometimes the required functionality is not present in the systems in use, preventing such actions to be made. Here I discuss the positive aspect of how Open Source software can be modified by users to implement what is missing, and how it can be done.

Implementing new functionality to create gateways for interoperability

The problem just described when implementing interoperability efforts is that the desired functionality behind it is sometimes not supported. One way to handle this is to find alternative ways to implement, but this can be

argued to be leading to non optimal conditions. Another way to tackle the problem of lacking functionality is to create the functionality. Normally when a software is proprietary, the company owning and in charge of the source code have to implement the functionality, then release a new update, and have the updates be installed before the functionality is readily available for the users. That is of course if they actually decide that it is a good idea to implement the functionality in the first place. If you're a smaller actor making up a small portion of the users of a system, it is naturally unlikely that a company will consider your wishes over the majority. If the software is Open Source on the other hand, anyone can make changes if they want to. As it is stated in [45] ".. open source software products have been and are being developed, distributed, and supported in the field on a voluntary basis by and for users themselves - no supplier required". There are however some challenges related to this, which is soon discussed.

Implementing new features to allow for system migration

As it was just discussed, the reduction of systems through migration can be a way to deal with fragmentation issues. This can be done by moving one or more systems' functionality into another, effectively creating one large system rather than having multiple smaller ones. For this to be possible however, the functionality of the system to be migrated must be present in the target system. If a system is proprietary and closed source, it can be challenging if not impossible to make changes to the code to implement such functionality. If the application is Open Source however, it is possible for anyone to make changes, or ask the surrounding community of the software project to implement the changes.

Challenges of actual implementation

When using Open Source software anyone can in theory implement the functionality they want as it is described by the authors of [22]. But although it is possible in theory, it can be challenging in practice. As the authors of [22] explain, you first of all need to have the technical know-how to program and change the code to create what you want to be supported. You need to be able to download the source code, identify what you want to be made, find a good way to do it, and then develop a working solution.

When you then want to make use of what has been made, there are two choices. One way is to add to the base of the software, making your changes available as a feature of the system for everyone to use. Since Open Source software projects are usually a collaboration between multiple people you cannot simply make changes to the code as you wish however. As it is explained by the authors of [30] "Even though participants are free to take the project where they want as long as they release the modified code, acceptance by the leadership of a modification or addition provides some certification as to its quality and its integration/compatibility with the overall project.". One example of how this works now follows. Open

Source software is usually stored in an online repository which anyone can access and download from. To make changes to the repository, you have to have the rights to do so. These rights are given to you by other developers on the same project who already have the power to modify. To gain such rights, you have to acquire the trust of the other members of the project who already have the power to change the code, and to administer rights. Even after gaining such rights, you often have to go through a process of agreement that a piece of code should be added to the software base. Then, once the code has been tested and pushed into the online repository, you can download and make use of the new version. In the case of DHIS 2, the developers with access to change the online repository was originally limited to people at the University of Oslo, but has now been expanded, and rights have been given to members across the world. At the time of writing there are 44 registered core developers and 330 registered developers[2].

The other way to use the made changes is to use them regardless of the rest of the development team. Nothing stops you from directly using the modified software you made, but by skipping the approvement process you will now be using a different version than everybody else. This excludes you from other updates made by the rest of the team. Doing this is called making a fork, or 'forking' [30][26], which means taking one software project, and breaking off from the process of the base, and starting your own competing project. You are free to upload your changes into a new repository, effectively starting a new collaboration project which people can add to, but if you are only making small changes and want the updates coming from others this is not the way to go. You could also keep using the base of the original code, and download new versions which you locally modify with your additions before using it, but it would become troublesome and impractical over time.

That said, developing and adding to Open Source software is easier said than done. In theory it's possible to modify the code to create the functionality you want, but it implies that you are a skilled software developer capable of understanding the problem and programming a working solution, before even thinking of publishing the code into the repository.

Another way to have additional functionality implemented into an Open Source software is through the existing project community. DHIS 2 has a big community of users and developers who are actively taking part in improving it every day. The way users participate can be as simple as asking questions about the use of the system, but it is also possible to request new features. It is common that experienced DHIS 2 users answer questions from less experienced users through the use of different mailing lists related to the project. When it comes to the particular problem of creating endpoints for creating gateways however, the actors involved will probably be more experienced in the field of software engineering. Hippel and Lakhani [28] explain how contributers are motivated to respond to user requests in that "software writing and debugging may in fact be motivated

[2]https://launchpad.net/dhis2

by personal benefit from the work product, by fun and learning associated with performing the work and by rep utational considerations".

One way for functionality to be implemented in an Open Source project like DHIS 2 is to register something called blueprints. Blueprints are a way of defining a task to be done and adding it to a list of tasks within the project context which the developers can access. Whenever such a blueprint has been defined, it will become a part of the future workload which the developers select from when they have time to take on new tasks. The problem here though is obviously that your blueprint has to appear to be more important than other ones if it is to be selected and finished by one of the developers. That might be challenging depending on the case, and the blueprints list can get flooded with requests. At the time of writing there were 292 registered in the DHIS 2 repository, where only a few of them are currently under development. The way to get your blueprint taken on is often achieved through discussions in the mailing lists, which anyone can join in on. By taking part in discussions you will get to know the other developers while they get to know you, while you at the same time get to tell people about your idea so that it might get picked up and finished.

Furthermore DHIS 2 have dedicated software developer teams who are working full time and are being paid to develop functionality. There are teams working at the University of Oslo, in Vietnam, in India, and in South Africa. This adds further to the motivation for the developers, and the possibility that a users functionality request will be implemented. Having paid developers working on Open Source software is not exclusive to DHIS 2, and in fact the authors of [37] explain that in the recent years, commercial companies have increased their involvement in such projects.

However, when it comes to developing functionality for creating gateways it is possible to argue that if you're going to change the code in the first place to develop an endpoint in which to make a gateway on top of, then why wouldn't you just solve the problem in some other way. If you are going to do that much work in the first place, perhaps the whole point of a gateway is redundant since it is meant to be a relatively simple way to connect systems with what's already available. On the other hand when it comes to larger systems, creating an endpoint in the form of a data export can be achieved with little effort compared to implementing the functionality of an otherwise working solution into another software. If you were using two systems performing very different tasks like for example one system for nationally storing personal information about citizens, and another system for tracking treatment of tuberculosis it would require substantially more work to replace one system into the other, rather than making it possible to link the two. You could then for example through some sort of gateway link a tuberculosis patient to a national ID in a register.

7.4.2 Open Source software lowering costs and encouraging local capacity

The software company Voxiva had been responsible for much of the computer based HIS development in Rwanda. Specifically they had

been delivering systems based on mobile phone reporting. The main systems were the HIV tracking system TracNet, and the Community Health Information System mUbuzima. The products were however proprietary and costly. At the time systems were being phased out, and were being replaced by other software systems, mostly by DHIS 2.

The replacement of the Voxiva based systems with DHIS 2 was a process which can be said to have contributed to sustainability in the Rwanda HIS through lower costs and through encouraging local capacity. Voxiva delivered their products as a service, claiming to be Software as a Service (SaaS) provider[3]. What that implies is that the provided software service is hosted, controlled and maintained by the company, while the subscribers simply use the system. That means that there are running costs of hosting and support and maintenance personnel, in addition to the cost of the software itself. In the paper [28] it is explained how this common, and show how Open Source software support on the other hand is usually given on a free and voluntary basis. The reasons why can perhaps be explained by the study done by Lakhani and Von Hippel [28] where the two top answers from participants in a user-to-user support community of an Open Source software when asked why they answered support questions was "I help now so I will be helped in the future ", and "I have been helped before so I reciprocate". Furthermore, the software itself is free[37]. Although it can be argued to be a good thing to be able to outsource such staff removing stress on the subscribers, the opposite could contribute to local capacity building. Additionally, at least in the TracNet case, the costs for recording data were high. Since the system was based on IVR technology, the CHW reporting the data would sit through calls that were usually more than 15 minutes each to report the data, due to the amount of data elements to be reported. The cost of such lengthy calls were high compared to other solutions such as sending a single SMS, which further added to the costliness of the system in the first place.

The way the new DHIS 2 based replacements of the systems are different is both in cost of the software itself, and in maintenance fees. It is also different in the way that everything is locally hosted and administered. One issue that was presented to me was that the HMIS staff were not able to directly access the TracNet data. To get the access they would have to call up the Voxiva people, who could then provide the desired data. One way to look at this is that it is both time-wasting and unecessary, if the HMIS people are able to run the system themselves.

For running DHIS 2 The MoH in Rwanda manage their own servers and databases out of their HMIS offices in central Kigali. Since the software is Free and Open Source, they were able to download and deploy on a local machine themselves, although with some initial support from HISP actors. Furthermore they were getting support through the DHIS 2 online communities, and reported that they in the beginning were only receiving help, but later helped other newcomers to the DHIS 2 scene. This was mostly done through mailing lists which anyone can subscribe to, free of

[3]https://www.voxiva.com/index.php/company/support

charge.

By running and maintaining everything themselves they were effectively reducing overhead timewaste, saving on costs, and building local capacity.

The opposite can be said to be reverse-capacity building. When everything is maintained by outside actors, the users are dependent on a company and are not able to learn how to maintain and administer the system themselves, and are not learning how to run their own systems independently.

Implementing SMS support in DHIS 2

Implementing SMS functionality in DHIS 2 in Rwanda was started because of the interest of using it to enable more reporting scenarios, and to replace other software systems such as the TracNet IVR. Actually setting it up and getting it to run has been proven a challenging task, as there are both technical and non-technical aspects that need to be completed before it can be used. The payoffs with getting the SMS support in DHIS 2 to work however are big because once they have the short code SMPP subscription set up, they could potentially relatively simply design as many reporting forms as they wanted in the future. At the moment of writing, health workers had to use a desktop computer with an internet connection to report data into DHIS 2. Usually they reported large monthly forms which were gathered from a collection of health workers in different areas or disciplines. Using the SMS functionality would enable for direct data entry in even the most remote areas with only needing a mobile handset with basic SMS functionality. It could potentially allow for more specific data entry, and more frequent reporting. This could for example result in more detailed data analysis as day to day statistics are more probable when all that is needed is sending a single SMS rather than sitting down and logging into a computer at a Health Center every day. Further it would also enable automatic SMS message notifications which could strengthen the HIS by for instance automatically monitoring for outbreaks and sending alerts to the appropriate actors to respond.

7.5 Reflection on my research approach

During the course of this thesis I based my work methods on the ideas of Action Research and Participatory Design. The Action Research literature presented ways of facilitating the research, which was beneficial in order to help structure, and focus on an area of appropriate size. A more direct method that was followed was Participatory Design, which was used within the framework defined based on the ideas of Action Research.

Working on the described projects at the Ministry of Health, and with the HMIS staff in Rwanda gave me valuable insight into the situations where the projects were taking place. Getting to know the participants, and the problems in question whilst working on a daily basis and following

the progress was an absolutely necessary part of it, which showed how Participatory Design is benificial in such efforts.

When it comes to the app development I was through the use of Participatory Design whilst being present in Rwanda, and by taking advantage of the situation where I worked closely with the HMIS on project work able to communicate first hand with some of the intended end users which helped respond to user needs even before the application was released. In [25] it is explained how "participatory design is regarded as an effective approach in systems development processes to overcome challenges such as changing contexts, difficulties of capturing users' needs and problems of achieving system acceptance". Some times it is hard to imagine how the user will actually use the product in the end, and getting feedback throughout the development process has shown just that. Some of the features that were implemented were a direct result of feedback from when the people at the Ministry of Health in Rwanda tried to use an unfinished version of the application, for example the ability to view Dashboard elements in a zoomable full screen mode.

A structured model of Participatory Design has not been used, but the related theory as explained by the authors of [38] has been the inspiration for the involvement of end users throughout the design and development processes. The use of the ideas from Participatory Design has rather taken the form of informal testing and communication with users of the existing DHIS 2 base. Rapid prototyping as explained in [9] has been a big part of the development process where sometimes even daily updates and feedback sessions have been the case.

Working alongside the main HMIS staff in the Ministry of Health in Rwanda has undoubtedly enabled following the principle of the bridging and blurring of the user-designer distinction as explained in [43] as one of the fundamental aspects of Participatory Design.

7.6 Summary of the research questions

Before concluding I look back at the four reserach topics presented in the introduction, and summarize what has been discussed related to them.

- By going to Rwanda and working with the HIS personnel I have been able to find out how the system works and have presented it in chapter 5. I got to understand who the different actors were, including the different ranges of users and computer systems. One of the key challenges is the lack of capacity, both in the financial aspect, and in the one of staff. The HIS is under constant development however, and as they are building their own capacity through training users and by reducing unecessary costs the HIS is becoming more sustainable.

- Using Open Source software has proved to be a successful approach to HIS computer software implementation. DHIS 2 has after 2-3 years been deployed country wide in Rwanda, and there were current processes taking place where other non-Open Source software systems

were being replaced by the Open Source DHIS 2 software. When the country were using proprietary Software as a Service solutions, everything was maintained by outside actors in foreign countries. The hosting and maintenance of the DHIS 2 is now done by local staff at the Ministry of Health which is building capacity locally, rather than in countries in Europe and North America. Since the Open Source software solutions are free of charge as opposed to the proprietary ones, it is more sustainable as it requires less funding.

- Challenges with fragmentations of computer systems leads to duplication and manual synchronization work. Methods of tackling such problems have been presented and discussed and it shows that using gateways to create interoperability engines can make manual efforts automatic. Another effort to address these problems that has been shown is that of reducing systems and merging them together. In the cases discussed in this thesis computer systems have been merged into systems based on the DHIS 2 software, which has also reduced the costs of licensing and maintenance as some of the systems that were merged into the free of charge DHIS 2 one were originally non-free.
- Making an Android application for easy access to analytical functionality of DHIS 2 has been proved applicable in the context of people in analytical positions. Being able to quickly view data directly from DHIS 2 databases without needing to access a web browser can increase data use. The use of Participatory Design through working alongside some of the intended end users has contributed to additional functionality and better usability from idea sharing, feedback and beta testing, and from working in and getting to know the day to day context in which the app can be used.

Chapter 8

Conclusion

Computer based systems are an important part of the HIS in Rwanda. Open Source software plays a big role, and the biggest systems are implemented using Open Source solutions. The increasing use of Open Source software moves more and more of the maintenance and administration efforts into the countries that are using the systems, which contributes to sustainability in having lower costs, and by enabling and encouraging the creation of local capacity. Local capacity is being built by that the local staff become experts in their areas and systems, which further enable them to train others with their own skills, and become independent of outside support. Additionally, Open Source softwares such as DHIS 2 have global communities that are easy to take part in where users can seek help and contribute by joining in workshops and mailing lists.

Another aspect of the Rwandan HIS is that it consists of many different computer systems that each serve their own purpose in closed environments. Some of the data within the systems are relevant to the other systems, and data is sometimes duplicated as the systems usually don't communicate with each other. This can result in redundant work in synchronization and because data is collected multiple times. Creating an application that automatically synchronizes systems can help with optimization and handling of such challenges. Such automization applications can be created in a relatively simple manner by using gateways such as exporting and importing functionality of the systems.

Lastly, during the development process of an Android application designed to be used in developing country HIS, the use of Participatory Design has proved beneficial. DHIS 2 is the biggest computer based system in Rwandas HIS and is deployed country-wide for purposes of aggregate data collection, patient tracking, and performance based financing reporting. Health workers throughout the country report data directly to a central database which allows for direct and immediate analysis of location based data for those in decision making positions. Making an Android application for handheld devices which enables quicker and easier access has proved to be applicable, and the effort has received positive feedback from other countries using the software as well as from users in Rwanda. Taking advantage of the situation of working alongside

the end users through the methods of Participatory Design has yielded results in the form of added functionality based on direct input during the development stage.

Bibliography

[1] Carla Abou-Zahr and Ties Boerma. 'Health information systems: the foundations of public health'. en. In: *Bulletin of the World Health Organization* 83 (2005), pp. 578–583.

[2] Ciborra et. al. *From Control to Drift: The Dynamics of Copropate Information Infrastructures*. Oxford University Press, 2000. ISBN: ISBN 0-19-829734-3.

[3] The World Bank. *Rwanda Overview*. URL: http://www.worldbank.org/en/country/rwanda/overview (visited on 19/04/2014).

[4] Jesús Bisbal et al. 'Legacy information systems: Issues and directions'. In: *IEEE software* 16.5 (1999), pp. 103–111.

[5] J. Bisbal et al. 'An overview of legacy information system migration'. In: *Software Engineering Conference, 1997. Asia Pacific ... and International Computer Science Conference 1997. APSEC '97 and ICSC '97. Proceedings*. Dec. 1997, pp. 529–530.

[6] Jørn Braa and M Humberto. 'Building collaborative networks in Africa on health information systems and open source software development–Experiences from the HISP/BEANISH network'. In: *IST Africa* 3 (2007).

[7] Jørn Braa, Eric Monteiro and Sundeep Sahay. 'Networks of action: sustainable health information systems across developing countries'. In: *Mis Quarterly* 28.3 (2004), pp. 337–362.

[8] Jørn Braa and Sundeep Sahay. *Integrated health information architecture : power to the users : design development and use*. New Dehli : Matrix Publishers, 2012, pp. 139–177. ISBN: 978-93-81320-06-8.

[9] Jørn Braa and Sundeep Sahay. *Participatory Design within the HISP network*. Routledge, 2012, pp. 235–256.

[10] Jørn Braa et al. 'Developing Health Information Systems in Developing Countries: The Flexible Standards Strategy'. In: *MIS Quarterly* 31.2 (2007), pages.

[11] Michael L. Brodie and Michael Stonebraker. *Migrating Legacy Systems: Gateways, Interfaces &Amp; the Incremental Approach*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1995. ISBN: 1-55860-330-1.

[12] Peter Checkland and Sue Holwell. ‘Action Research: Its Nature and Validity’. In: *Systemic Practice and Action Research* 11.1 (1998), pp. 9–21.

[13] Jared Cohen. *One hundred days of silence: America and the Rwanda Genocide*. Rowman & Littlefield, 2007, p. 1. ISBN: 978-0-7425-5237-1.

[14] Robert Davison, Maris G. Martinsons and Ned Kock. ‘Principles of canonical action research’. In: *Information Systems Journal* 14.1 (2004), pp. 65–86. (Visited on 01/03/2014).

[15] Fiona Godlee et al. ‘Can we achieve health information for all by 2015?’ In: *The Lancet* 364.9430 (2004), pp. 295–300.

[16] Judith Gregory. ‘Scandinavian approaches to participatory design’. In: *International Journal of Engineering Education* 19.1 (2003), pp. 62–74.

[17] Ole Hanseth. ‘Gateways-just as important as standards: How the internet won the "religious war" over standards in Scandinavia’. In: *Knowledge, Technology & Policy* 14.3 (2001), pp. 71–89.

[18] Ole Hanseth and Eric Monteiro. ‘Inscribing behaviour in information infrastructure standards’. In: *Accounting, Management and Information Technologies* 7.4 (1997), pp. 183–211.

[19] Ole Hanseth, Eric Monteiro and Morten Hatling. ‘Developing Information Infrastructure: The Tension Between Standardization and Flexibility’. In: *Science, Technology & Human Values* 21.4 (1996), pp. 407–426.

[20] Wilhelm Hasselbring. ‘Information System Integration’. In: *Commun. ACM* 43.6 (2000), pp. 32–38.

[21] Richard Heeks. ‘Information Systems and Developing Countries: Failure, Success, and Local Improvisations’. In: *The Information Society* 18.2 (2002), pp. 101–112.

[22] Guido Hertel, Sven Niedner and Stefanie Herrmann. ‘Motivation of software developers in Open Source projects: an Internet-based survey of contributors to the Linux kernel’. In: *Research policy* 32.7 (2003), pp. 1159–1177.

[23] Edoardo Jacucci, Vincent Shaw and Jørn Braa. ‘Standardization of health information systems in South Africa: The challenge of local sustainability’. In: *Information Technology for Development* 12.3 (2006), pp. 225–239.

[24] Jembi. *Jembi RHEA Project Wiki*. URL: https://jembiprojects.jira.com/wiki/display/RHEAPILOT/Home (visited on 06/03/2014).

[25] Honest Christopher Kimaro and Ola Hodne Titlestad. ‘Challenges of user participation in the design of a computer based system: The possibility of participatory customisation in low income countries’. In: *Journal of Health Informatics in Developing Countries* 2.1 (2008).

[26] Bruce Kogut and Anca Metiu. 'Open-source software development and distributed innovation'. In: *Oxford Review of Economic Policy* 17.2 (2001), pp. 248–264.

[27] Magnus Korvald. 'Integrating mobile and web health infrastructures in low resource contexts'. MA thesis. University of Oslo, 2013.

[28] Karim R Lakhani and Eric Von Hippel. 'How open source software works: "free" user-to-user assistance'. In: *Research policy* 32.6 (2003), pp. 923–943.

[29] W.M. Lee. *Beginning Android 4 Application Development*. ITPro collection. Wiley, 2012. ISBN: 9781118240670.

[30] Josh Lerner and Jean Tirole. 'Some simple economics of open source'. In: *The journal of industrial economics* 50.2 (2002), pp. 197–234.

[31] Theo Lippeveld, Rainer Sauerborn and Claude Bodart. *Design and implementation of health information systems*. World Health Organization, 2000, pp. 3–5. ISBN: 92-4-156199-8.

[32] S. K. Lwanga, Cho-Yook Tye and O. Ayeni. 'Teaching Health Statistics'. In: (1999), pp. 23–24. URL: http://libdoc.who.int/publications/1999/9241545186_eng_part1.pdf (visited on 12/02/2014).

[33] Jean McNiff. *Action Research: principles and practice*. Routledge, 2013, pp. 23, 89. ISBN: ISBN 9780203112755.

[34] Jinsoo Park and Sudha Ram. 'Information Systems Interoperability: What Lies Beneath?' In: *ACM Trans. Inf. Syst.* 22.4 (2004), pp. 595–598.

[35] The Linux Information Project. *Proprietary Software Definition*. URL: http://www.linfo.org/proprietary.html (visited on 10/02/2014).

[36] Inc Red Hat. *What is open source?* URL: http://opensource.com/resources/what-open-source (visited on 24/11/2013).

[37] Jeffrey A Roberts, Il-Horn Hann and Sandra A Slaughter. 'Understanding the motivations, participation, and performance of open source software developers: A longitudinal study of the Apache projects'. In: *Management science* 52.7 (2006), pp. 984–999.

[38] Toni Robertson and Jesper Simonsen. *Beginning Android 4 Application Development*. Routledge, 2013. ISBN: 978-0415720212.

[39] Government of Rwanda. *Brief History of Rwanda*. URL: http://www.gov.rw/History (visited on 24/11/2013).

[40] Noam Shemton and Ian Walden. *Free and Open Source Software: Policy, Law and Practice*. Oxford University Press, 2013, pp. 20–27. ISBN: ISBN 9780199680498.

[41] Amit P. Sheth. 'Changing Focus on Interoperability in Information Systems:From System, Syntax, Structure to Semantics'. In: *Interoperating Geographic Information Systems*. Ed. by Michael Goodchild et al. Vol. 495. The Springer International Series in Engineering and Computer Science. Springer US, 1999, pp. 5–29. ISBN: 978-1-4613-7363-6.

[42] Pandula Anilpriya Siribaddana and Sundeep Sahay. 'Integrating blended-learning for health information systems training in developing countries: Towards a conceptual framework'. In: (2013).

[43] Ola Hodne Titlestad, Knut Staring and Jørn Braa. 'Distributed Development to Enable User Participation.' In: *Scandinavian Journal of Information Systems* 21.1 (2009).

[44] USAID. *History of USAID/Rwanda*. URL: http://www.usaid.gov/history-usaidrwanda (visited on 19/04/2014).

[45] Eric Von Hippel. 'Open source software projects as user innovation networks'. In: *Open Source Software Economics* (2002).

[46] Steven Weber. *The Success of Open Source*. Harvard University Press, 2005. ISBN: 9780674018587.

[47] Alfred Winter et al. *Health Information Systems : Architectures and Strategies*. Springer, 2011, pp. 26–33. ISBN: 9781849964418.

[48] Bing Wu et al. 'Legacy system migration: A legacy data migration engine'. In: *Proceedings of the 17th International Database Conference*. 1997, pp. 129–138.

[49] Bing Wu et al. 'The Butterfly Methodology: a gateway-free approach for migrating legacy information systems'. In: *Engineering of Complex Computer Systems, 1997. Proceedings., Third IEEE International Conference on*. Sept. 1997, pp. 200–205.